

AI VIETNAM
All-in-One Course
(TA Session)

Poem Generation Project



AI VIET NAM
[@aivietnam.edu.vn](http://aivietnam.edu.vn)

Dinh-Thang Duong - TA

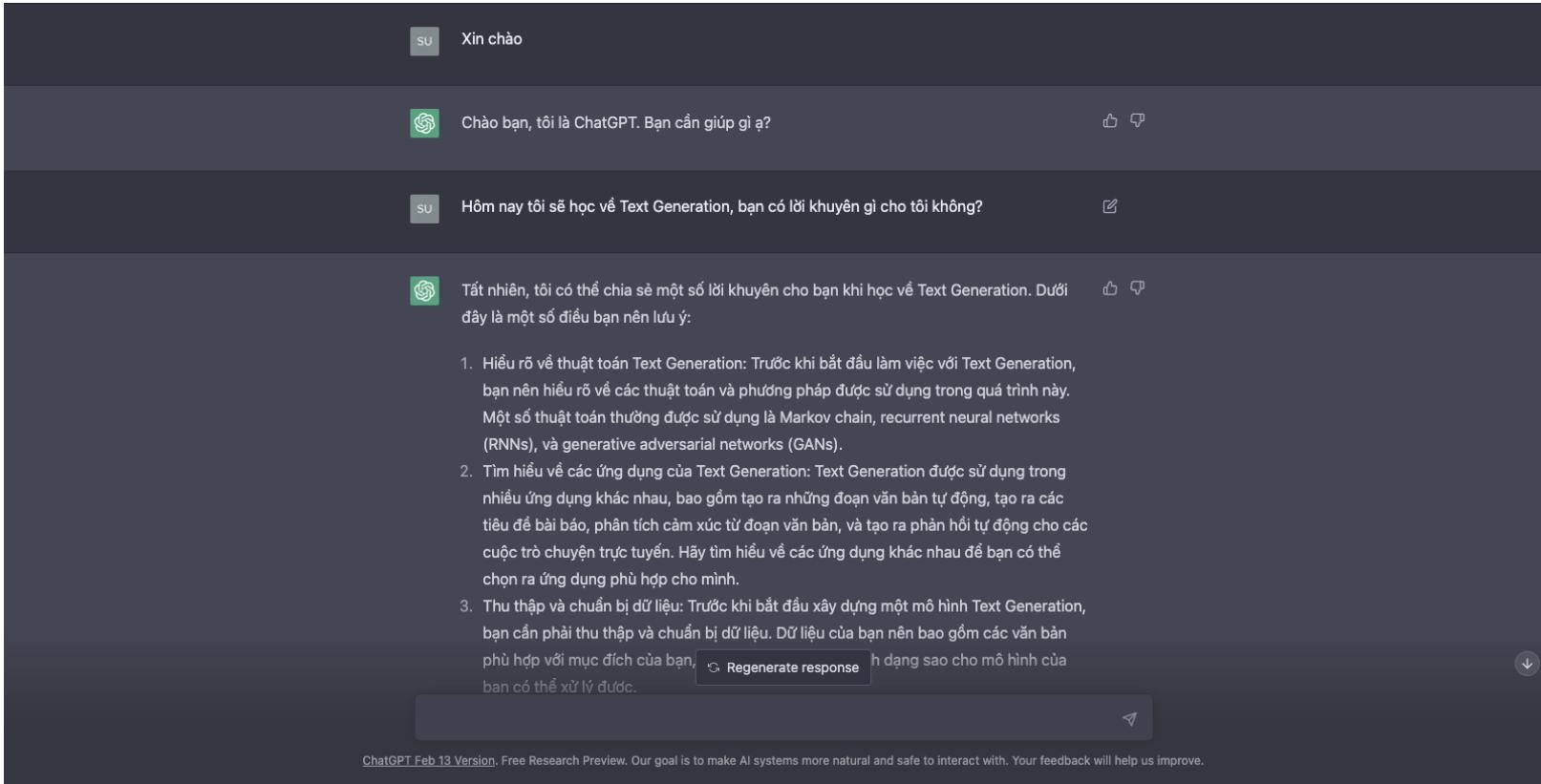
Outline

- Introduction
- Poem Data Crawling
- Poem Generation: From Scratch
- Poem Generation: GPT2
- Question

Introduction

Introduction

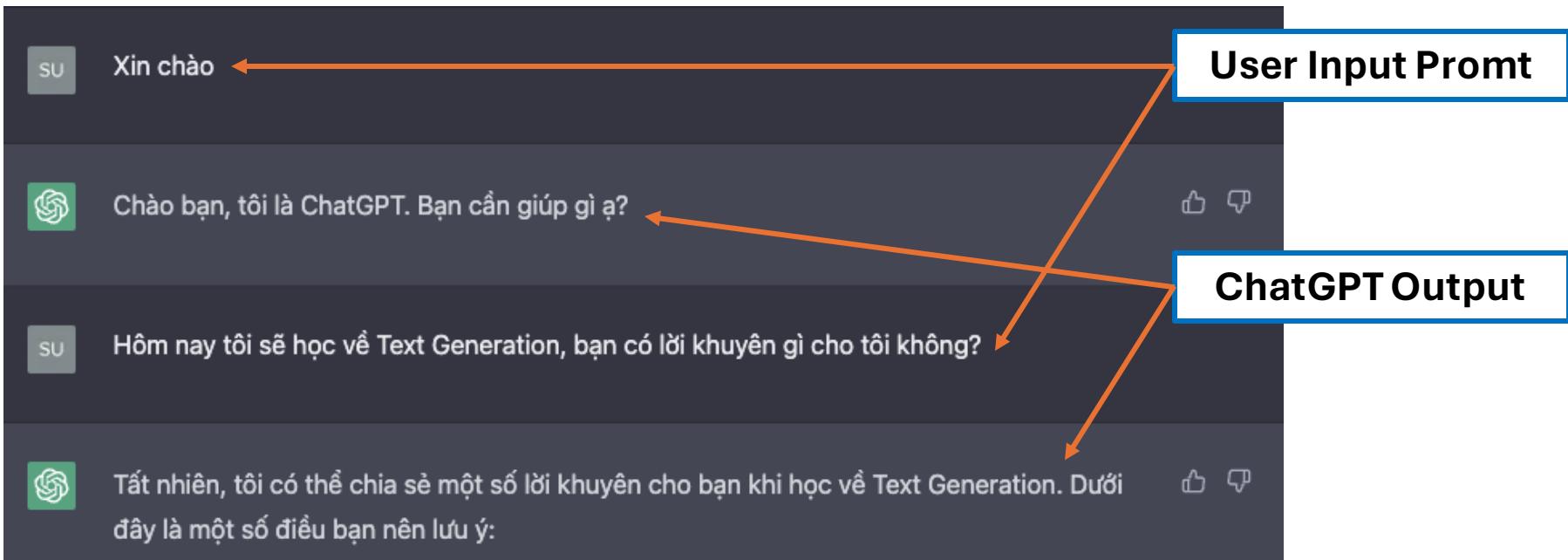
❖ Getting Started



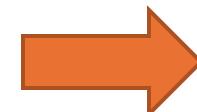
ChatGPT - The most famous AI application at the moment

Introduction

❖ Getting Started



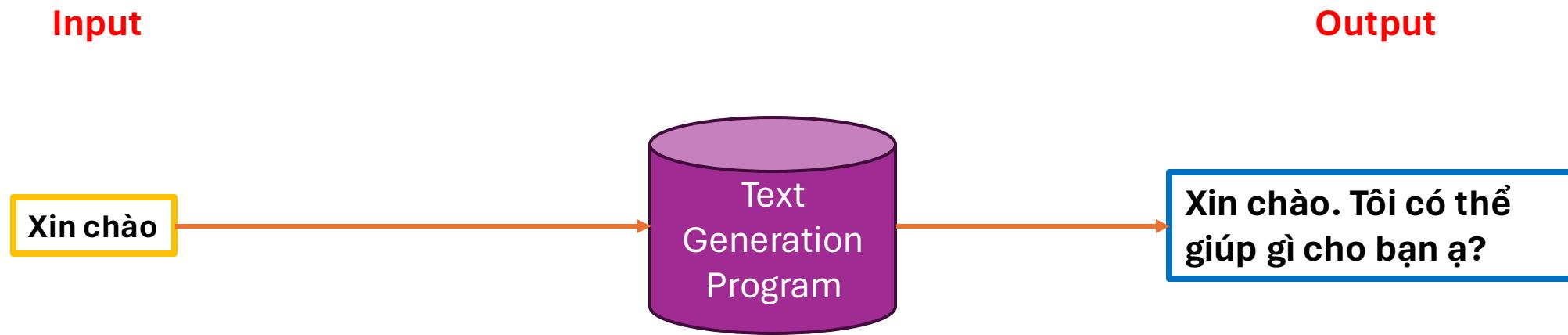
ChatGPT takes user input text and generate an appropriate response



Text Generation

Introduction

❖ Definition



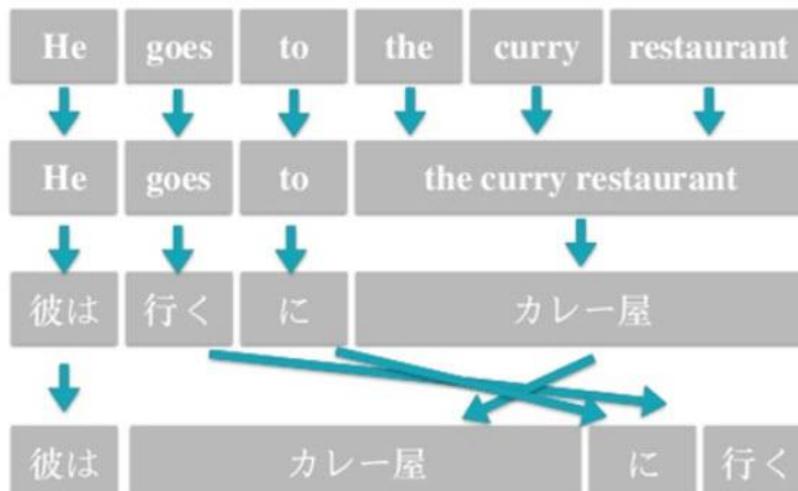
Text Generation: An NLP task that aims to automatically generate new text based on some inputs.

Introduction

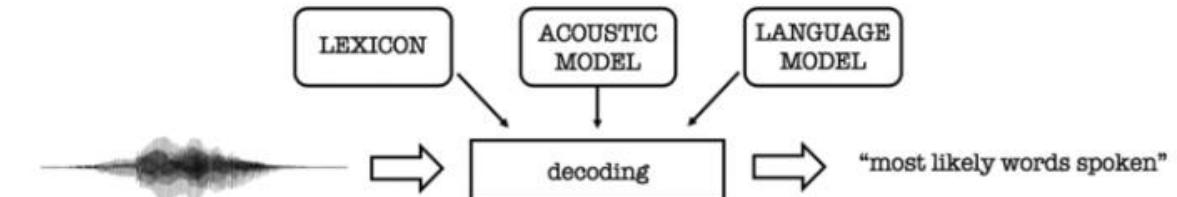
❖ Text Generation Applications



Image Captioning



Machine Translation



Automatic Speech Recognition

Source Text: Peter and Elizabeth took a taxi to attend the night party in the city.

While in the party, Elizabeth collapsed and was rushed to the hospital.

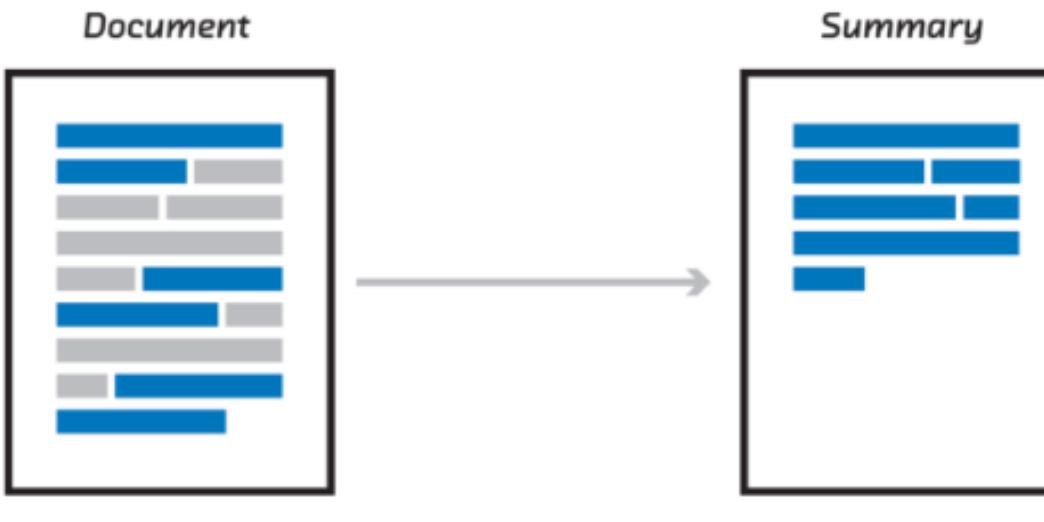
Summary: Elizabeth was hospitalized after attending a party with Peter.



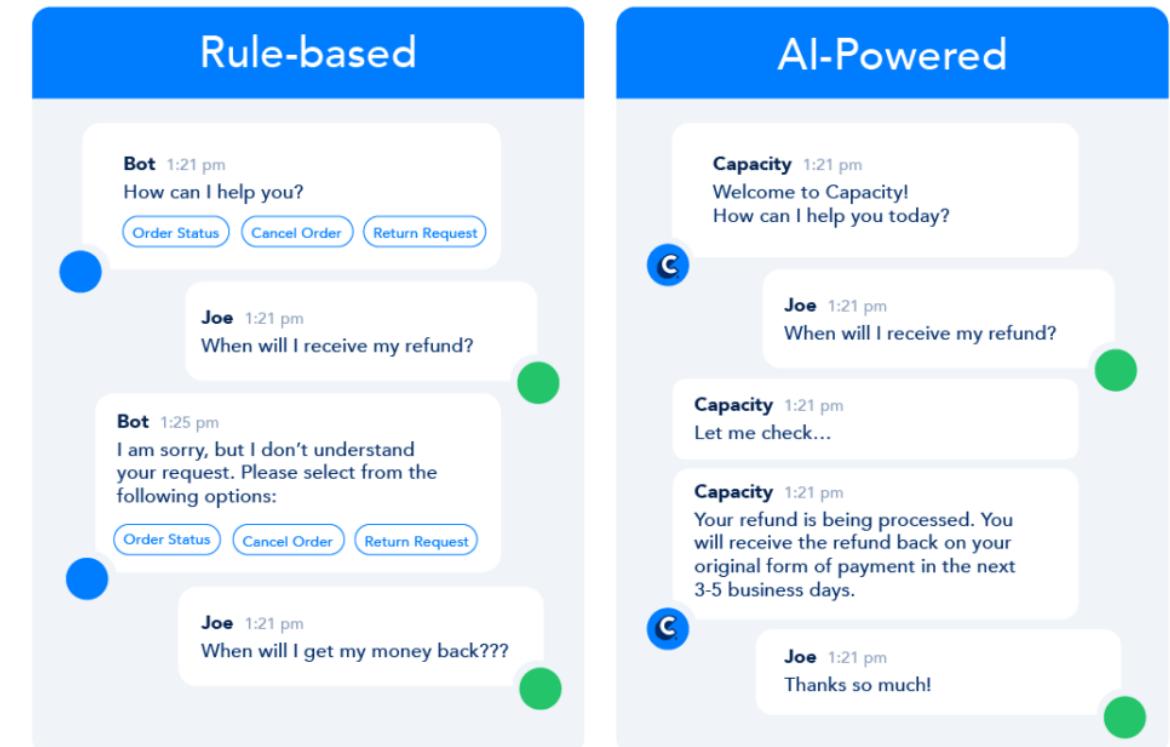
Abstractive Text Summarization

Introduction

❖ Text Generation Applications



Extractive Text Summarization



Rule-based Chatbot vs Text Generation Chatbot

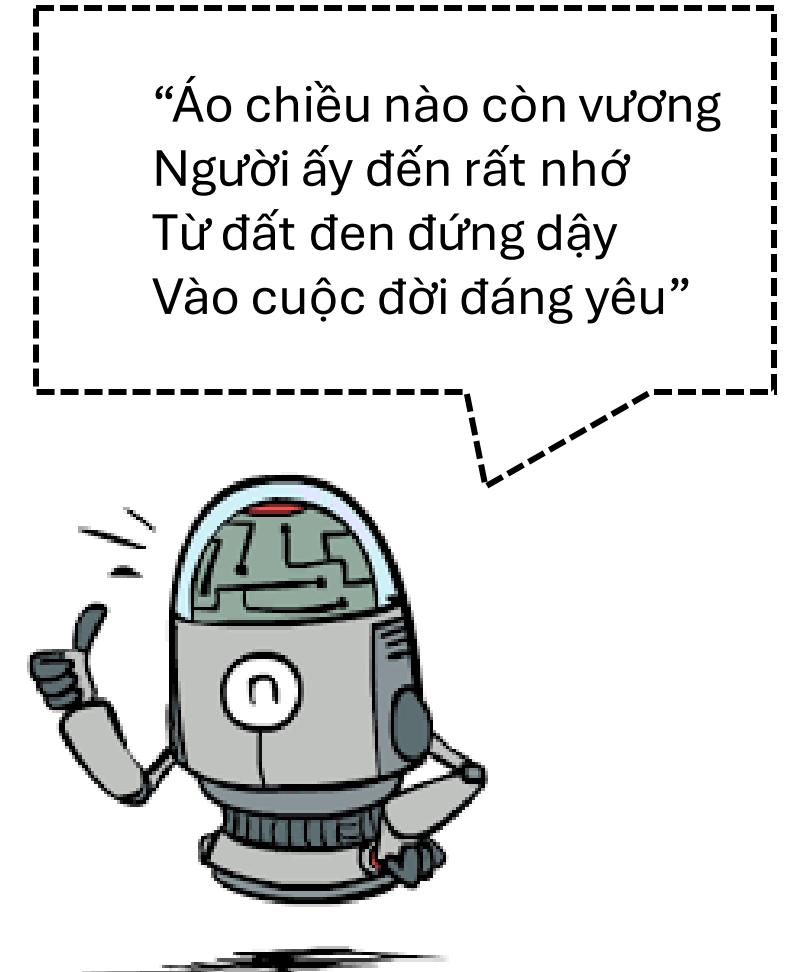
Text Generation refers to the task of generating new words

Introduction

❖ Project Description



Project:
Vietnamese Poem
Generation



Introduction

❖ Getting Started

Project Description: Given a Vietnamese poem line or some words, generate a poem.

Input

Promt: “Áo chiều nào còn vương”



Output

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

Introduction

❖ Project Pipeline

Project Description: Given a Vietnamese poem line or some words, generate a poem.

Input

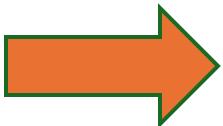
Promt: “Áo chiều nào còn vương”



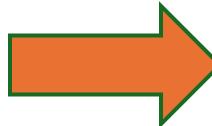
Output

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

1. Collecting Dataset



2. Prepare Training Data



3. Train and evaluate model



4. Writing inference program

Poem Data Crawling

Introduction

❖ Project Pipeline

Project Description: Given a Vietnamese poem line or some words, generate a poem.

Input

Promt: “Áo chiều nào còn vương”



Output

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

1. Collecting Dataset

2. Prepare Training Data

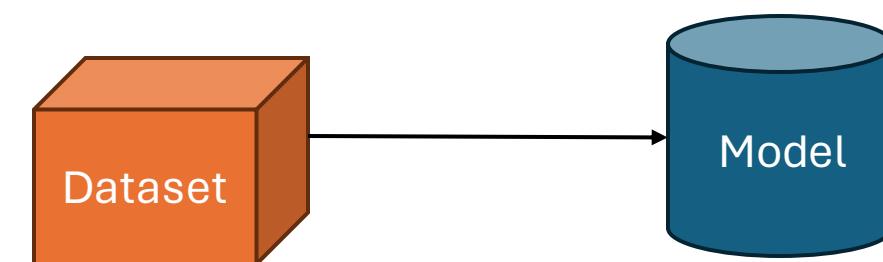
3. Train and evaluate model

4. Writing inference program

Poem Data Crawling

❖ Problem

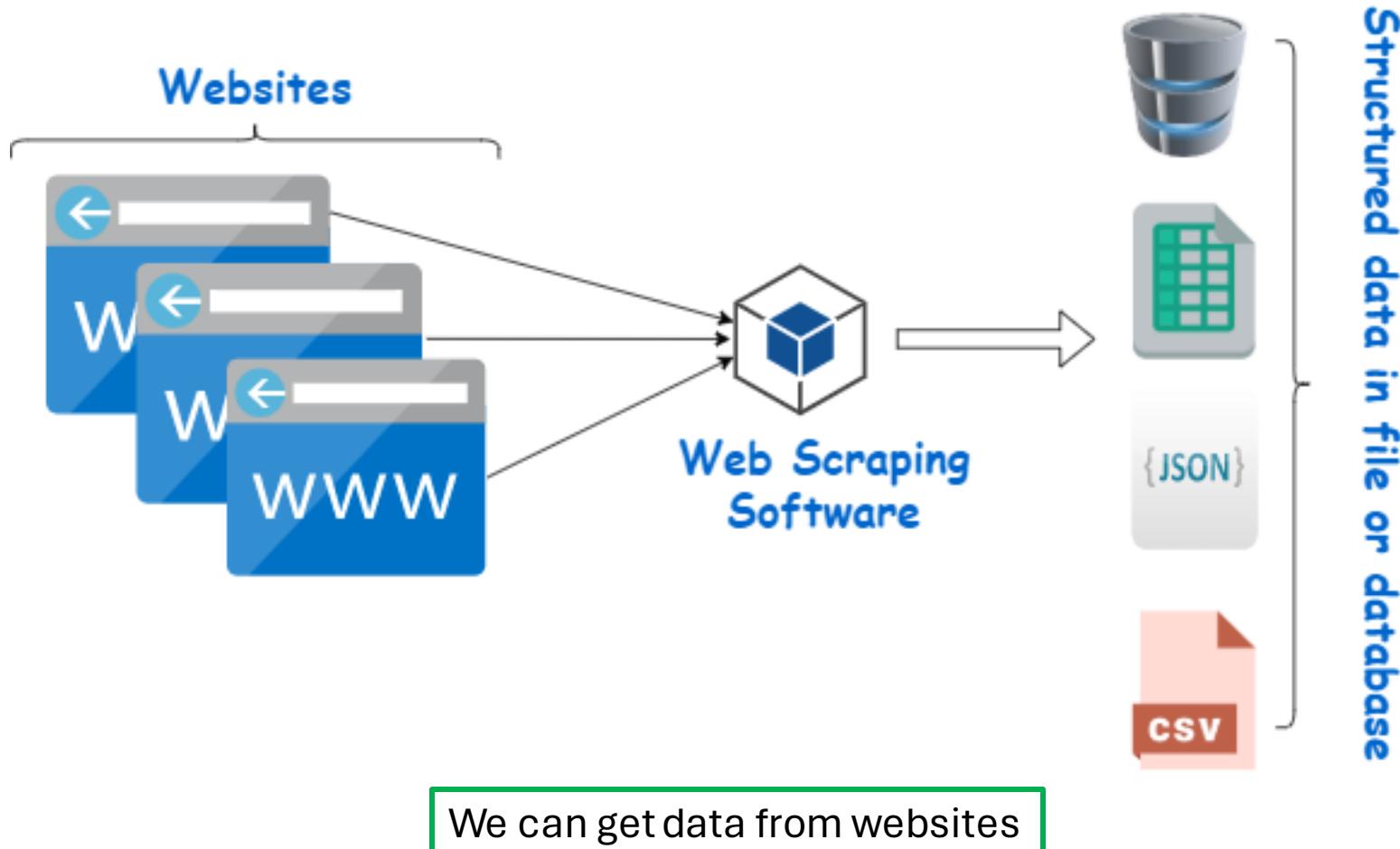
	title	content
0	“Bạn xấu như chiếc bóng”	Bạn xấu như chiếc bóng\nCứ bám riết theo anh...
1	“Cái làm ta hạnh phúc”	Cái làm ta hạnh phúc\nThực ra cũng chẳng nhiều...
2	“Chiều vừa xốp trên tay”	Chiều vừa xốp trên tay\nChợt nghe thoảng ong b...
3	“Chơi thân không có nghĩa”	Chơi thân không có nghĩa\nKhông cãi nhau bao g...
4	“Có thể buồn chút ít”	Có thể buồn chút ít\nMột mình, không người yêu...
...
95	Ám ảnh sông xưa	Ôi, con sóng chết khô,\nvật vờ trong bùn quanh...
96	Áng dương không biết sầu	Áng dương không biết sầu\nNằm mãi ở trên cao\n...
97	Anh	Cây bút gãy trong tay\nCení mực khô đáy lọ\nÁnh...
98	Anh biết	Không có anh để già\nLàm sao em được trẻ\nMuốn...
99	Anh bốn mùa lập đồng	Buồn, thì buồn thiệt đó\nEm à, có biết không\n...



We don't have data yet

Poem Data Crawling

❖ Solution: Data Scraping



Poem Data Crawling

❖ Introduction to Web Scraping

The screenshot shows a website interface for 'THI VIỆN'. At the top, there's a navigation bar with social media icons (Facebook, Twitter, Pinterest, Tumblr) and a 'Follow' button. Below the navigation, a search bar displays '93.063 bài thơ, 4.761 tác giả [Alt+3]' and a 'Loạn' (Random) button. To the right are tabs for 'TÁC GIẢ', 'THƠ', 'THAM GIA', 'KHÁC', and a highlighted 'Đăng nhập' (Login) button.

Nội dung chính

- Thư viện thơ: 93.063 tác phẩm của 4.761 tác giả từ 104 nước, trong đó
 - 63.271 bài tiếng Việt
 - 17.207 bài chữ Hán
 - 12.585 bài ngôn ngữ khác
- Thơ thành viên: 38.071 bài của 1.521 tác giả
- Diễn đàn: 3.559 chủ đề
- Nội dung chi tiết theo các chuyên mục...

Trích dẫn

“*Cây cúc đắng quên lòng
mình đang đắng,
Trò hoa vàng đọc suối để
ong bay.*”⁹⁹

-- Đi trong rừng (Phạm Tiến Duật)

Tác giả mới

- Đỗ Anh Thư (Việt Nam)
- Marius Chelaru (Rumani)
- Eric Patrick Clapton (Anh)
- Alain Delorme (Pháp)
- Lionel Brockman Richie (Mỹ)

Thơ Việt mới

- Bên chiều ru nhở (Minh Sơn Lê)
- Chuyện tình Quasimodo (Đỗ

thivien.net

Poem Data Crawling

❖ Introduction to Web Scraping

Trang trong tổng số 521 trang (5203 bài thơ)

Xếp theo: ▾

[1] [2] [3] [4] ... Trang sau » Trang cuối

"Cái làm ta hạnh phúc"

★★★★★ 4x4.75

Việt Nam » Hiện đại » Thái Bá Tân » Châm ngôn mới (2013)

0 trả lời, 2190 lượt xem, 0 người thích

Do hongha83 gửi ngày 21/07/2021 14:40

"Chiều vừa xốp trên tay"

★★★★☆ 2x3.00

Việt Nam » Hiện đại » Lâm Huy Nhuận » Chiều có thật (1999)

0 trả lời, 2453 lượt xem, 1 người thích

Do hongha83 gửi ngày 27/12/2013 18:43

"Dưới giàn hoa thiên lý..."

★★★★★ 4x5.00

Việt Nam » Hiện đại » Nguyễn Nhật Ánh » Mắt biếc (1990)

0 trả lời, 6548 lượt xem, 0 người thích

Do tôn tiền tử gửi ngày 09/03/2015 22:02

"Đến, nhiều nơi để đến"

★★★★☆ 1x3.00

Việt Nam » Hiện đại » Thái Bá Tân » Châm ngôn mới (2013)

0 trả lời, 895 lượt xem, 0 người thích

Do hongha83 gửi ngày 21/07/2021 14:29

"Cái làm ta hạnh phúc"

Thơ » Việt Nam » Hiện đại » Thái Bá Tân » Châm ngôn mới (2013)

Cái làm ta hạnh phúc
Thực ra cũng chẳng nhiều
Chỉ cần có ai đó
Để ta thăm thương yêu

Rồi thêm chút công việc
Cho ta làm hàng ngày
Cuối cùng, chút mơ mộng
Để đưa ta lên mây



Nguồn: Châm ngôn mới (thơ), Thái Bá Tân, NXB Lao động, 2013

★★★★★

4 đánh giá, trung bình 4.75

Thể thơ: Thơ mới năm chữ
Thời kỳ: Hiện đại

- Chia sẻ trên Facebook
- Trả lời
- In bài thơ

Một số bài cùng tác giả

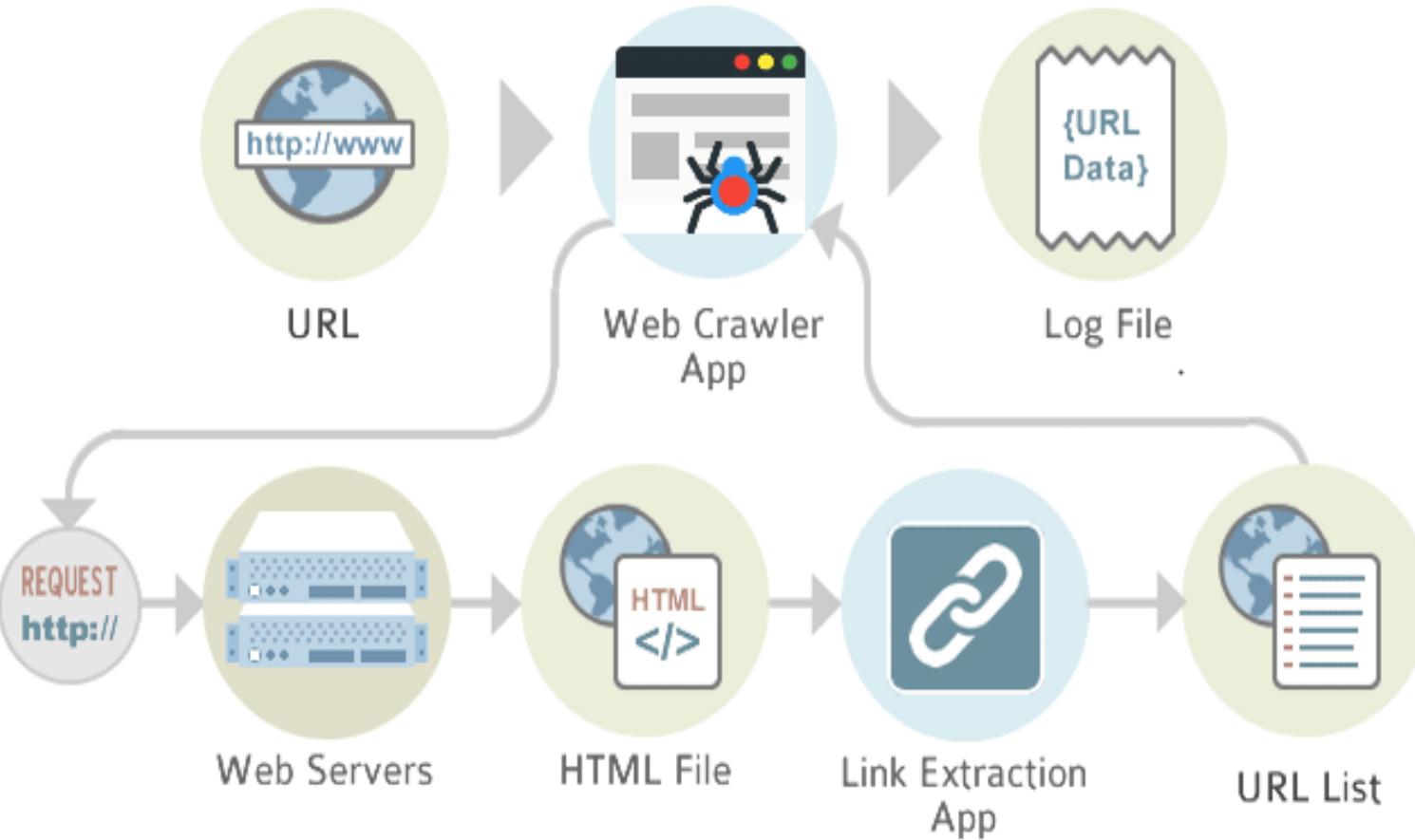
- Hai con dê qua cầu
- Con cáo và chùm nho
- Cái bàn
- Rùa và thỏ
- Khi mẹ mệt

Đăng bởi hongha83 vào 21/07/2021 14:40

We can collect pentagram poems on this website

Poem Data Crawling

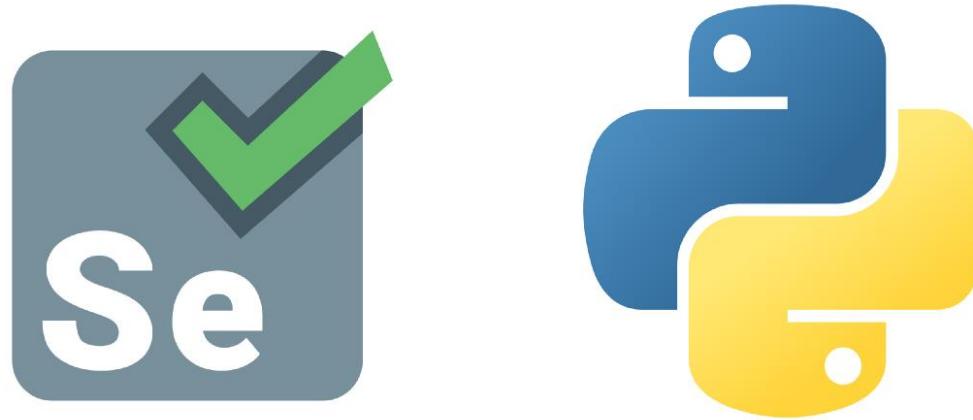
❖ Introduction to Web Scraping



Web Scraping: Extracting data from websites

Poem Data Crawling

❖ Selenium Package



Selenium Package: used to automate web browser interaction from Python.

Poem Data Crawling

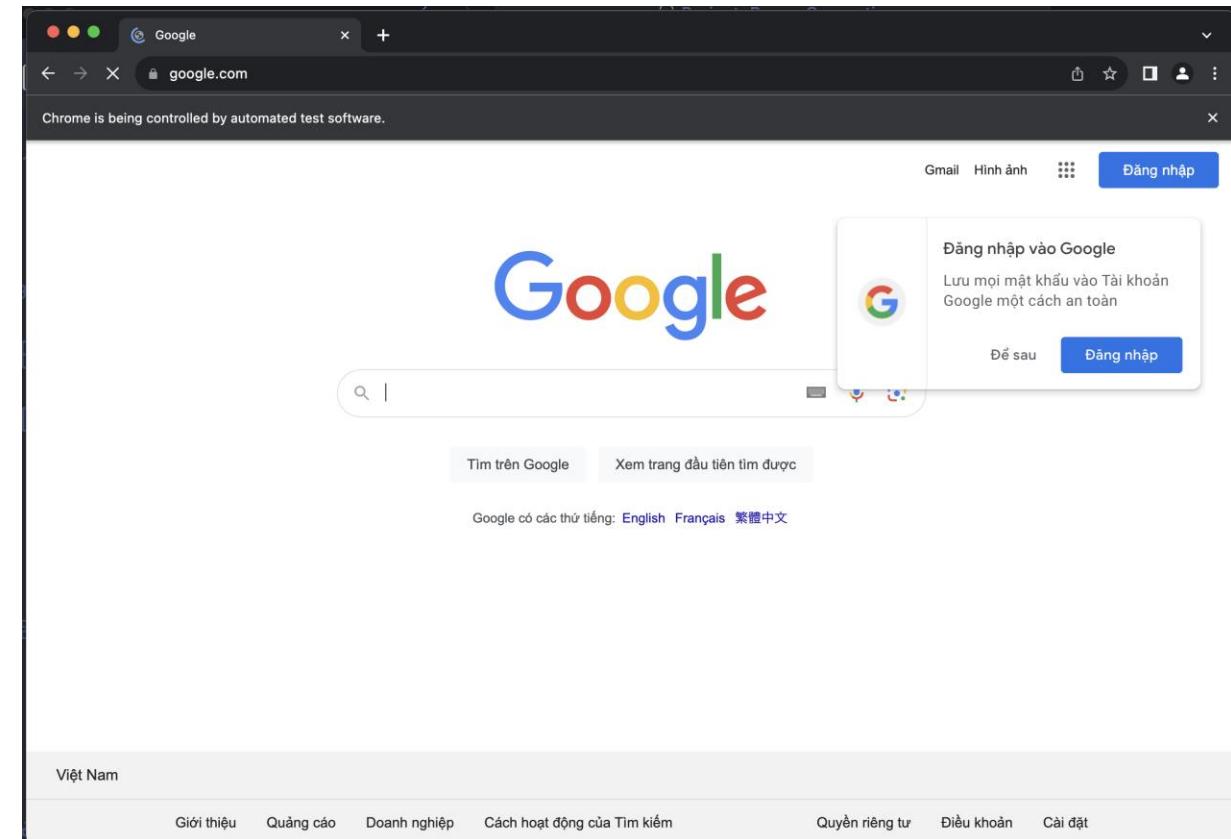
❖ Introduction to Selenium



Poem Data Crawling

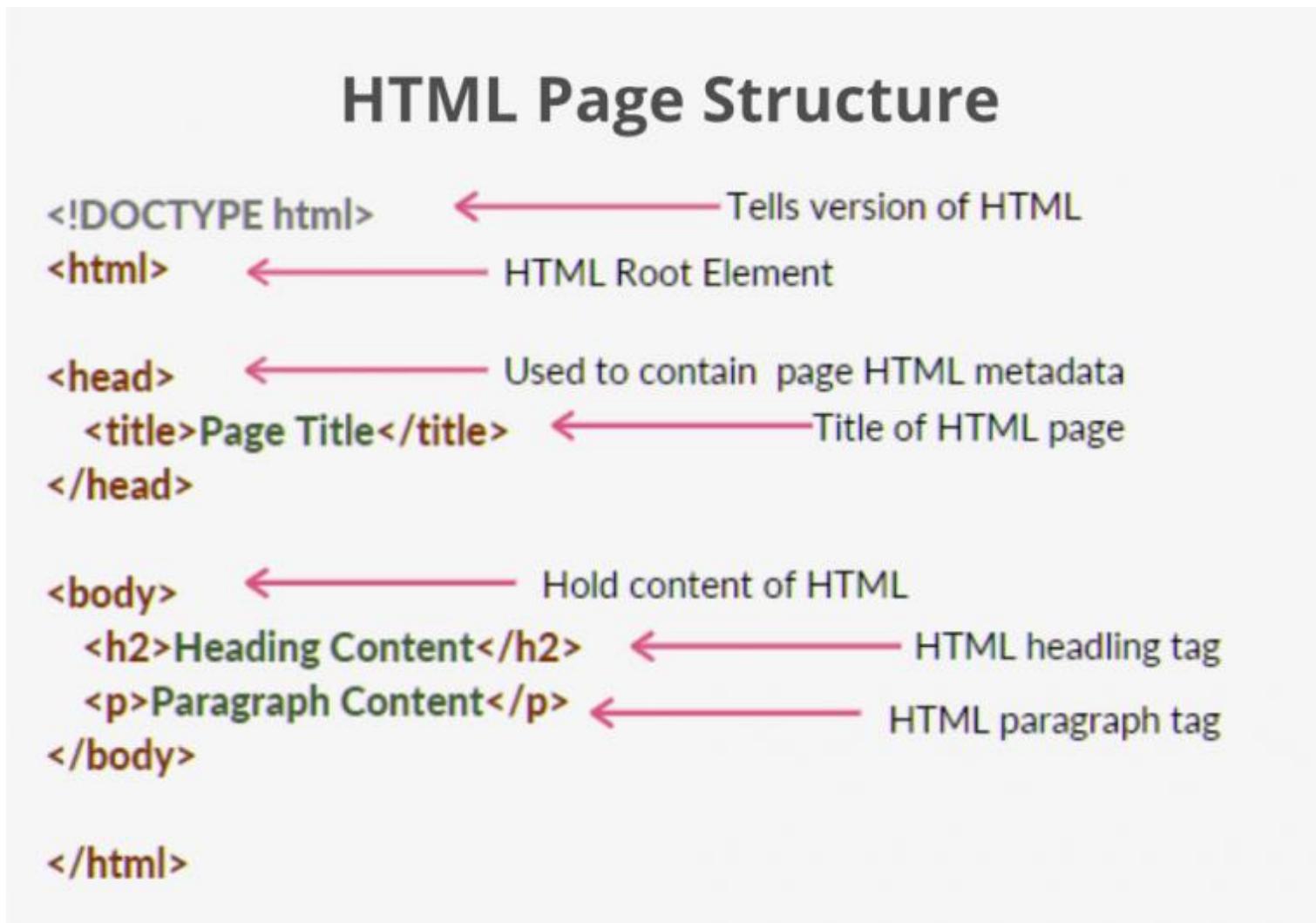
❖ Introduction to Selenium: Initialize driver

```
1 from selenium import webdriver
2
3 url = 'http://www.google.com'
4
5 driver = webdriver.Chrome()
6 driver.get(url)
```



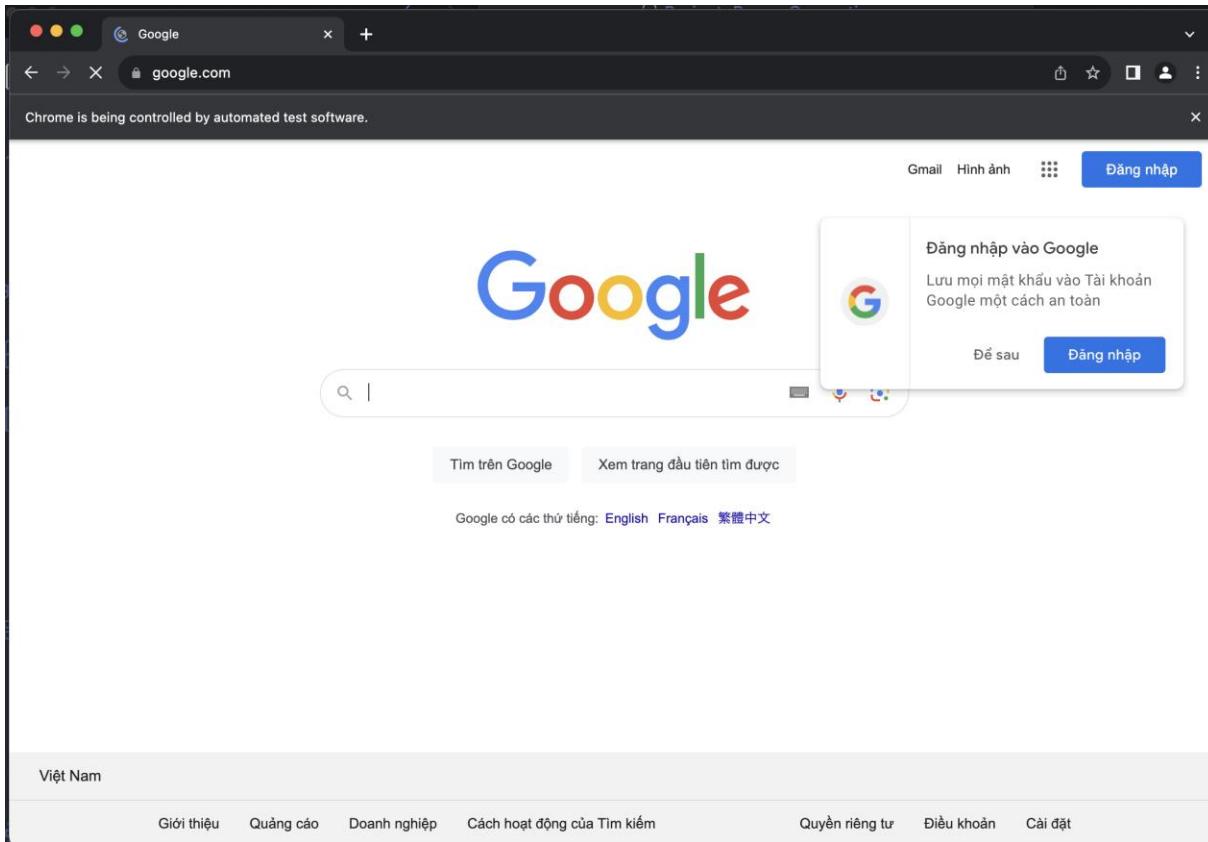
Poem Data Crawling

❖ Introduction to Selenium: HTML



Poem Data Crawling

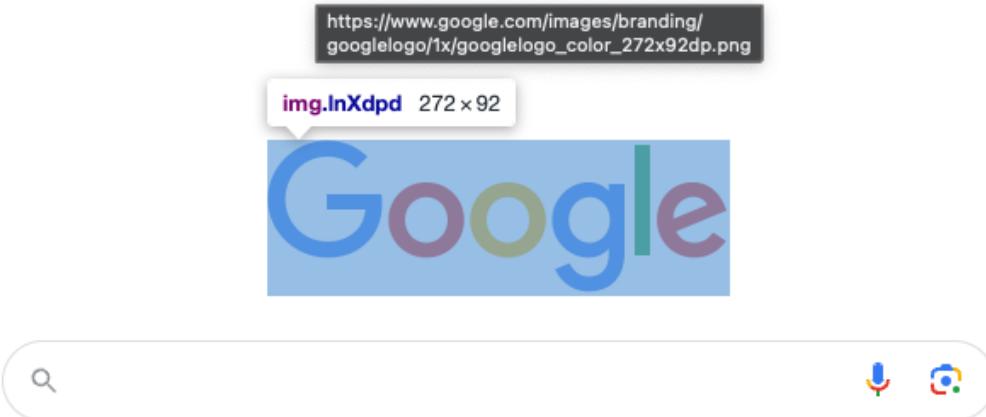
❖ Introduction to Selenium: HTML



```
<!DOCTYPE html>
<html itemscope itemtype="http://schema.org/WebPage" lang="en-VN">
  <head> ...</head>
  <body jsmode="hspDDf" jsaction="xjhTIf..CLIENT;02vyse..CLIENT;IVKTfe..CLIENT;Ez7VmC..CLIENT;YUC7He..CLIENT;hWT9Jb..CLIENT;WcUlWe..CLIENT;VM8bg..CLIENT;qqf0n..CLIENT;A8708b..CLIENT;Ycfj..CLIENT;szjOR..CLIENT;JL9QDc..CLIENT;kWlxhc..CLIENT;qGMTif..CLIENT;ydZCDF..CLIENT" data-new-gr-c-s-loaded="14.1156.0" data-new-gr-c-s-check-loaded="14.1156.0" data-gr-ext-installed>
    <style> ...</style>
    <div class="L3eUgb" data-hveid="1"> ...</div>
    <div class="o3j99 n1xJcf Ne6nSd" role="navigation"> ...</div> = $0
    <div class="o3j99 LLD4me yr19Zb LS80J"> ...</div>
    <div class="o3j99 ikrT4e om7nvf"> ...</div>
    <div class="o3j99 qarstb"> ...</div>
    <div jscontroller="B2qIPe" jsname="cgsKlb" jsaction="rcuQ6b:npT2md"> ...</div>
    <div class="o3j99 c93Gbe" role="contentinfo"> ...</div>
  </div>
  <div class="Fvgjic"> ...</div>
  <textarea class="csi" name="csi" style="display:none"></textarea>
  <div class="gb_jd" ng-non-bindable>Google apps</div>
  <div class="gb_q" ng-non-bindable>...</div>
  <script nonce="..."></script>
  <div id="sZmt3b"> ...</div>
  <script src="/xjs/_/js/k=xjs.hd.en.HYPqwJ8ZaHQ.0/ck=xjs.hd.uDeKTR1vBjE.L.W.0/am=AA..Y0x.m.s4mZb.mu.pFsdhd.phXghd,g0xTif,s39S4,s0XFj,gb_wiz,sf_sonic,spch?xjs=s1" nonce async gapi_processed="true"></script>
  <div id="snbc"> ...</div>
  <link href="/xjs/_/js/k=xjs.hd.en.HYPqwJ8ZaHQ.0/am=AAAAAAAAAAAAAAAQAAAATAQ..yn7.s.yN8,syn9,synb,DPreE,WlNQGd,syt0,syt2,nabPbb,kQylef,syt1,fX00xe?xjs=s3" rel="preload" as="script">
  <script src="/xjs/_/js/k=xjs.hd.en.HYPqwJ8ZaHQ.0/am=AAAAAAAAAAAAAAAQAAAATAQ..yn7.s.yN8,syn9,synb,DPreE,WlNQGd,syt0,syt2,nabPbb,kQylef,syt1,fX00xe?xjs=s3" nonce async>
  <link href="/xjs/_/js/k=xjs.hd.en.HYPqwJ8ZaHQ.0/am=AAAAAAAAAAAAAAAQAAAATAQ..=0/dg=2;br=1/rs=ACT90oHNKvfGpKEdMU7WvMtECne0tNw08A/m=syek,aLUfp?xjs=s3" rel="preload" as="script">
  <script src="/xjs/_/js/k=xjs.hd.en.HYPqwJ8ZaHQ.0/am=AAAAAAAAAAAAAAAQAAAATAQ..=0/dg=2;br=1/rs=ACT90oHNKvfGpKEdMU7WvMtECne0tNw08A/m=syek,aLUfp?xjs=s3" nonce async>
  <link href="/xjs/_/js/k=xjs.hd.en.HYPqwJ8ZaHQ.0/am=AAAAAAAAAAAAAAAQAAAATAQ..=0/dg=2;br=1/rs=ACT90oHNKvfGpKEdMU7WvMtECne0tNw08A/m=kMFpHd,sy8f,bm51tf?xjs=s3" rel="preload" as="script">
  <script src="/xjs/_/js/k=xjs.hd.en.HYPqwJ8ZaHQ.0/am=AAAAAAAAAAAAAAAQAAAATAQ..=0/dg=1/rs=ACT90oHNKvfGpKEdMU7WvMtECne0tNw08A/m=kMFpHd,sy8f,bm51tf?xjs=s3" nonce async>
  <script> ...</script>
  <iframe id="hfcr" src="https://accounts.google.com/RotateCookiesPage?og_pid=538&rot=3&origin=https%3A%2F%2Fwww.google.com&exp_id=0" style="display: none;"> ...</iframe>
</body>
<grammarly-desktop-integration data-grammarly-shadow-root="true"> ...</grammarly-desktop-integration>
</html>
```

Poem Data Crawling

❖ Introduction to Selenium: HTML Tag



Google Search

I'm Feeling Lucky

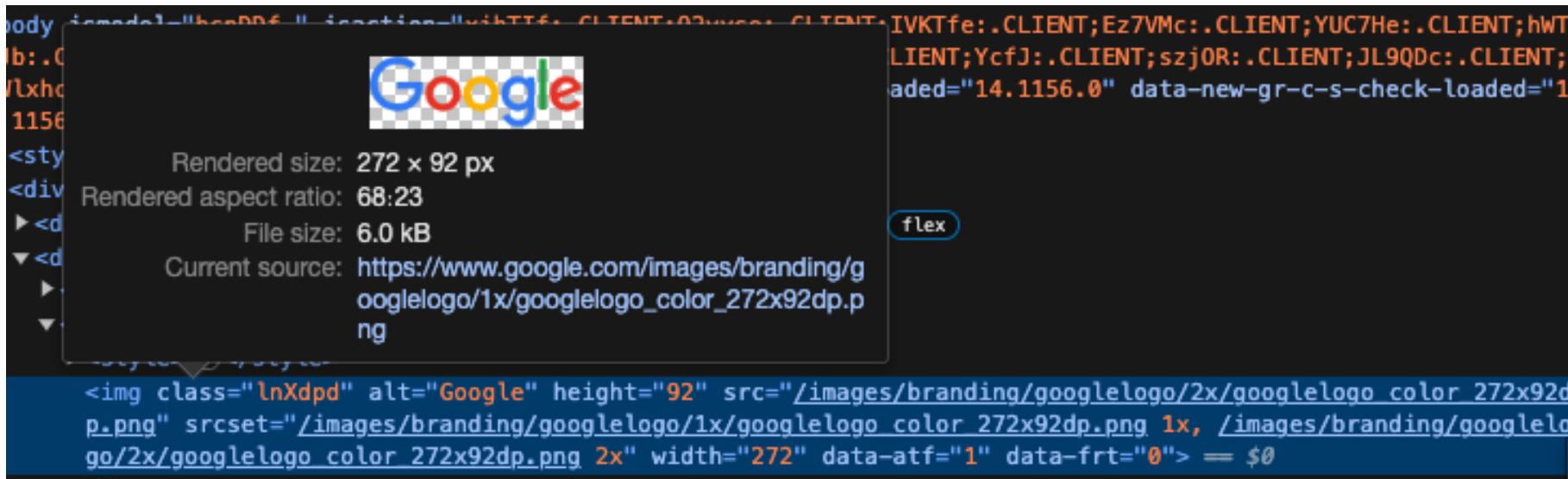
Google offered in: Tiếng Việt Français 繁體中文

```
▼<body jsmodel="hspDDf " jsaction="xjhTIf:.CLIENT;02vyse:.CLIENT;IVKTfe:.CLIENT;Ez7VMc:.CLIENT;YUC7He:.CLIENT;hWT9Jb:.CLIENT;WCulWe:.CLIENT;VM8bg:.CLIENT;qqf0n:.CLIENT;A870b:.CLIENT;T;YcfJ:.CLIENT;L09Pb:.CLIENT;L00Qb:.CLIENT;Lw1ub:.CLIENT;LmMTf:.CLIENT;ydZCDf:.CLIENT" data-r="instal" ▶<sty> ▶<div> Rendered size: 272 x 92 px ▶<d> Rendered aspect ratio: 68:23 ▶<d> File size: 6.0 kB ▶<d> Current source: https://www.google.com/images/branding/googlelogo/1x/googlelogo_color_272x92dp.png ▶<img alt="Google logo" data-bbox="354 111 508 178" data-gr-ext="flex" data-atf="1" data-frt="0" srcset="/images/branding/googlelogo/1x/googlelogo_color_272x92dp.png 1x, /images/branding/googlelogo/2x/googlelogo_color_272x92dp.png 2x" width="272" data-atf="1" data-frt="0"/> == $0 ▶</div> ▶</div> ▶<div class="o3j99 ikrT4e om7nvf">...</div> ▶<div class="o3j99 qarstb">...</div> ▶<div jscontroller="B2qlPe" jsname="cgsKlb" jsaction="rcuQ6b:npT2md">...</div> ▶<div class="o3j99 c93Gbe" role="contentinfo">...</div> ▶<div class="Fvgjgc">...</div> <textarea class="csi" name="csi" style="display:none"></textarea> <div class="gb_jd" ng-non-bindable>Google apps</div> ▶<div class="gb_q" ng-non-bindable>...</div> ▶<script nonce=>...</script> ▶<div id="Smt3b">...</div> <script src="/xjs/_/js/k=xjs.hd.en.HYPqwJ8ZaHQ.O/ck=xjs.hd.uDeKTR1vbjE.L.W.O/am=AA_Y0x.m54mZb.mu.pFsdhd.pHXqhd.q0xTif.s39S4.s0XFj.sb.wiz.sf.sonic.spch?xjs=s1" nonce async> noni processed="true">...</script>
```

This tag represents Google image

Poem Data Crawling

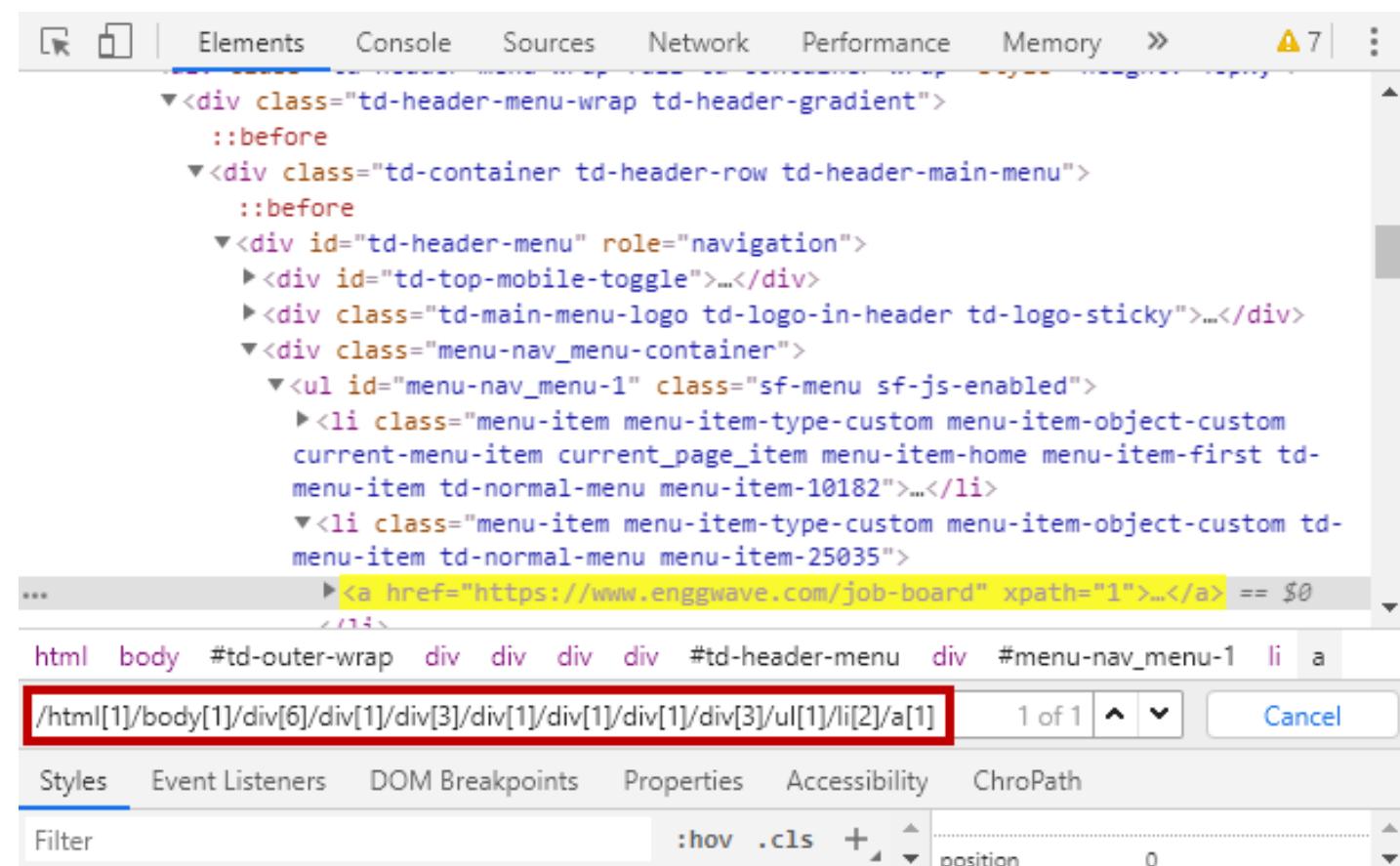
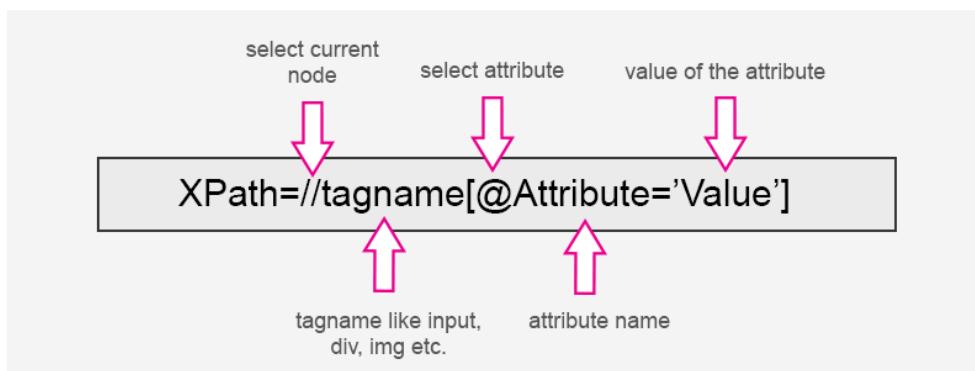
❖ Introduction to Selenium: HTML Tag



 tag contains src attribute, which contains image's url => How to find ?

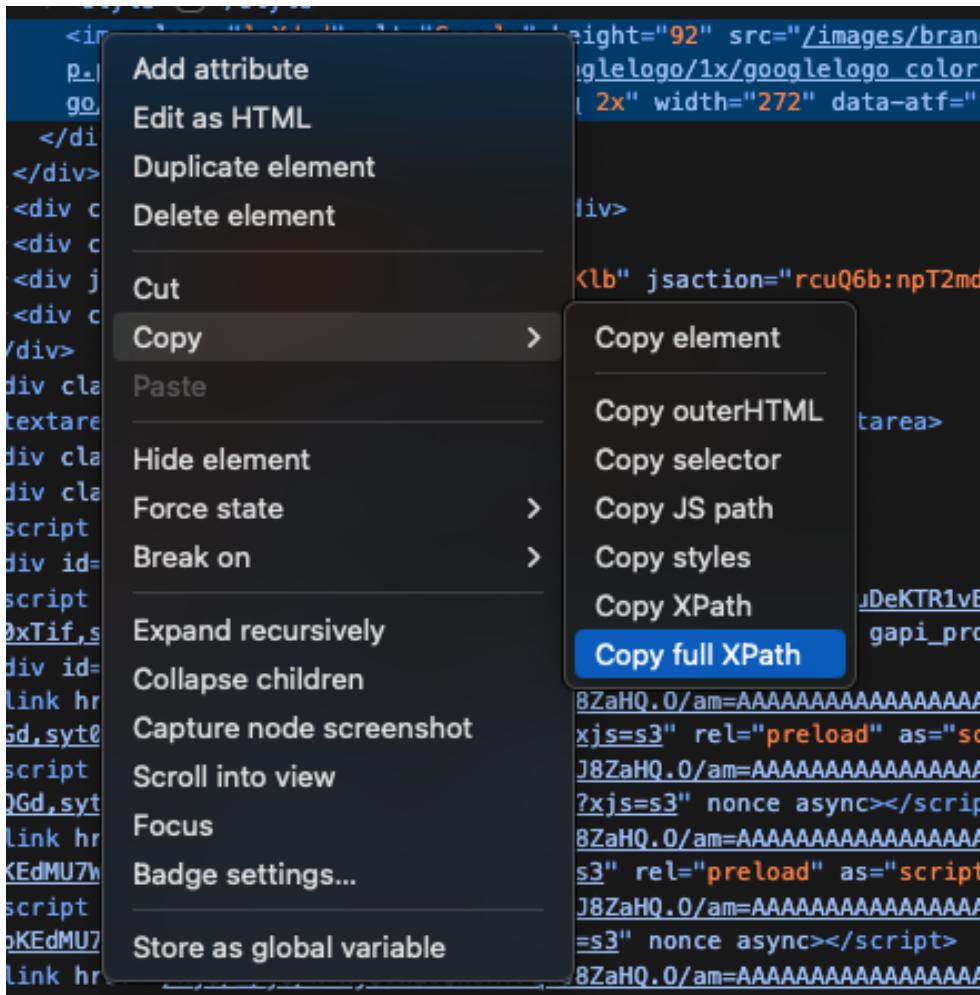
Poem Data Crawling

❖ Introduction to Selenium: HTML Tag



Poem Data Crawling

❖ Introduction to Selenium: HTML Tag



XPATH of tag: /html/body/div[1]/div[2]/div/img

Poem Data Crawling

❖ Introduction to Selenium: Extract Google image url

```
1  from selenium import webdriver
2  from selenium.webdriver.common.by import By
3
4
5  url = 'http://www.google.com'
6
7  driver = webdriver.Chrome()
8  driver.get(url)
9
10 img_url = driver.find_element(
11     By.XPATH,
12     '/html/body/div[1]/div[2]/div/img'
13 ).get_attribute('src')
14
15 print(img_url)
```

```
● (project_api_env) thangduong@Duongs-MacBook-Pro Selenium_Intro % python3 main.py
https://www.google.com/images/branding/googlelogo/2x/googlelogo_color_272x92dp.png
```



Poem Data Crawling

❖ Step-by-step: Identify target data location

The screenshot shows a web page titled "THI VIỆN". At the top, there are social media sharing icons and a message indicating 15K people are following the page. Below the header is a search bar containing the query "93.057 bài thơ, 4.761 tác/dịch giả [Alt+3]" and a "Loan" button. To the right of the search bar are navigation tabs: TÁC GIẢ, THƠ (which is currently selected), THAM GIA, KHÁC, and Đăng nhập. A dropdown menu titled "Các chuyên mục" is open, with "Tìm thơ..." highlighted by a red circle. Other options in the dropdown include: Thơ Việt Nam, Cổ thi Việt Nam, Thơ Việt Nam hiện đại, Thơ Trung Quốc, Đường thi, Thơ Đường luật, Tổng từ, and Thêm bài thơ... Below the dropdown, a large blue box contains the heading "TÌM BÀI THƠ:" and instructions: "Kết quả tìm được thỏa mãn **đồng thời** **tất cả** các tiêu chí bạn chọn. Bạn có thể tìm bằng Google với giao diện đơn giản hơn." There are two input fields labeled "Tiêu đề:" and "Tác giả:".

Poem Data Crawling

❖ Step-by-step: Identify target data location

TÌM BÀI THƠ:

Kết quả tìm được thoả mãn **đồng thời tất cả** các tiêu chí bạn chọn.
Bạn có thể tìm bằng [Google](#) với giao diện đơn giản hơn.

Tiêu đề:

Tác giả:

Nội dung:

Nguyên bản (thơ chữ Hán/tiếng nước ngoài):

Thể thơ:

Ngôn ngữ:

Nước:

Thời đại:

Poem Data Crawling

❖ Step-by-step: Identify target data location

1. Đường dẫn đến trang web

thivien.net/searchpoem.php?PoemType=16&Page=2

Tìm thơ
(thể loại **Thơ mới năm chữ**)

Trang **2** trong tổng số 521 trang (5203 bài thơ)
Xếp theo: **Tên bài**

"Không thể kiểm tra bạn"
Việt Nam » Hiện đại » Thái Bá Tân » Châm ngôn mới (2013)
0 trả lời, 468 lượt xem, 0 người thích
Do [hongha83](#) gửi ngày 22/07/2021 14:30

"Lặng ngồi chốn tĩnh lâu"
★★★★☆ 1x4.00
Việt Nam » Hiện đại » Thích Thiền Tâm » Niệm Phật thập yếu (2009)
0 trả lời, 2317 lượt xem, 0 người thích
Do [hongha83](#) gửi ngày 19/07/2017 07:11

"Lòng tôi là cây khế..."
★★★★★ 4x5.00
Việt Nam » Hiện đại » Nguyễn Nhật Ánh » Mắt biếc (1990)
0 trả lời, 6862 lượt xem, 1 người thích
Do [tôn tiên tử](#) gửi ngày 10/03/2015 22:04

2. Danh sách các bài thơ

Poem Data Crawling

❖ Step-by-step: Identify target data location

"Không thể kiểm tra bạn"

Việt Nam » Hiện đại » Thái Bá Tân » Châm ngôn mới (2013)

0 trả lời, 468 lượt xem, 0 người thích

Do hongha83 gửi ngày 22/07/2021 14:30

```
▼<div class="list-item">
  ▼<h4 class="list-item-header"> == $0
    <a href="/Th%C3%A1i-B%C3%A1-T%C3%A2n/Kh%C3%B4ng-th%C3%A9-kiem-tra-ban">"Không thể kiểm tra bạn"</a>
  </h4>
  ▶<div class="list-item-detail small">...</div>
</div>
```

"Không thể kiểm tra bạn"

Thơ » Việt Nam » Hiện đại » Thái Bá Tân » Châm ngôn mới

Không thể kiểm tra bạn
Như ta kiểm tra tiền
Muốn xem thật hay giả
Đem soi trước bóng đèn

Nguồn: *Châm ngôn mới* (thơ), Thái Bá Tân,
NXB Lao động, 2013



Poem Data Crawling

❖ Coding Step 1: Install selenium on Google 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
```

Poem Data Crawling

❖ Coding Step 2: Import libraries

```
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
```

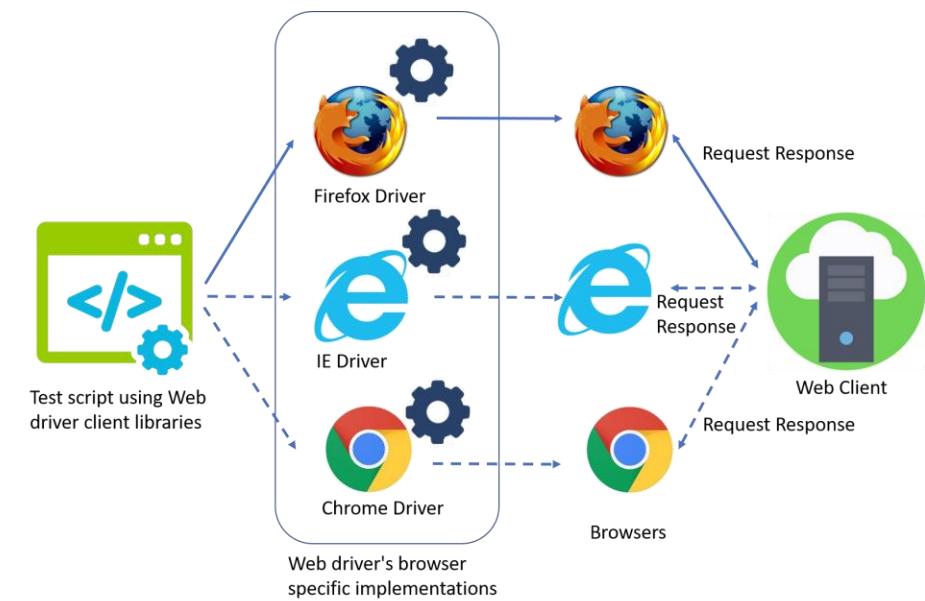


Selenium

Poem Data Crawling

❖ Coding Step 3: Create driver

```
1 WEBDRIVER_DELAY_TIME_INT = 10
2
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.headless = True
8 driver = webdriver.Chrome(service=service, options=chrome_options)
9 driver.implicitly_wait(5)
10 wait = WebDriverWait(driver, WEBDRIVER_DELAY_TIME_INT)
```



Poem Data Crawling

❖ Coding Step 4: Access to a page and get contents

```
1 datasets = []
2 deletion_script = 'arguments[0].parentNode.removeChild(arguments[0]);'
3 for page_idx in tqdm(range(1, 11)):
4     main_url = f'https://www.thivien.net/searchpoem.php?PoemType=1&ViewType=1&Country=2&Age []=3&Page={page_idx}'
5     driver.get(main_url)
6
7     content_tags_xpath = '//*[@class="page-content container"]/div[2]/div/div[@class="list-item"]'
8     content_tags = driver.find_elements(By.XPATH, content_tags_xpath)
```

"Cái làm ta hạnh phúc"

★★★★★ 4x4.75

Việt Nam » Hiện đại » Thái Bá Tân » Châm ngôn mới (2013)

0 trả lời, 2191 lượt xem, 0 người thích

Do hongha83 gửi ngày 21/07/2021 14:40

"Chiều vừa xốp trên tay"

★★★☆☆ 2x3.00

Việt Nam » Hiện đại » Lâm Huy Nhuận » Chiều có thật (1999)

0 trả lời, 2453 lượt xem, 1 người thích

Do hongha83 gửi ngày 27/12/2013 18:43

"Dưới giàn hoa thiên lý..."

★★★★★ 4x5.00

Việt Nam » Hiện đại » Nguyễn Nhật Ánh » Mắt biếc (1990)

0 trả lời, 6548 lượt xem, 0 người thích

Do tôn tiền tử gửi ngày 09/03/2015 22:02

Poem Data Crawling

❖ Coding Step 5: Take Poem Data

```
10 |     for idx in range(len(content_tags)):
11 |         content_tag_xpath = f'//html/body/div[4]/div[2]/div/div[{2+idx}]'
12 |         content_title_xpath = f'//html/body/div[4]/div[2]/div/div[{2+idx}]/h4/a'
13 |         content_tag = wait.until(EC.presence_of_element_located((By.XPATH, content_tag_xpath)))
14 |         poem_title = wait.until(EC.presence_of_element_located((By.XPATH, content_title_xpath))).text
15 |         poem_url = wait.until(EC.presence_of_element_located((By.XPATH, content_title_xpath))).get_attribute('href')
```



Poem Data Crawling

❖ Coding Step 5: Take Poem Data

```
17 try:  
18     driver.get(poem_url)  
19  
20     poem_src_xpath = '//div[@class="small"]'  
21     poem_content_tag = wait.until(EC.presence_of_element_located((By.CLASS_NAME, 'poem-content')))   
22  
23     try:  
24         poem_content_i_tag = poem_content_tag.find_element(By.TAG_NAME, 'i')  
25         driver.execute_script(deletion_script, poem_content_i_tag)  
26     except:  
27         pass  
28  
29     try:  
30         poem_content_b_tag = poem_content_tag.find_element(By.TAG_NAME, 'b')  
31         driver.execute_script(deletion_script, poem_content_b_tag)  
32     except:  
33         pass  
34  
35     poem_content = poem_content_tag.text  
36  
37     try:  
38         poem_src_tag = wait.until(EC.presence_of_element_located((By.XPATH, poem_src_xpath)))  
39         poem_src = poem_src_tag.text  
40     except:  
41         poem_src = ''
```

“Cái làm ta hạnh phúc”

[Thơ » Việt Nam » Hiện đại » Thái Bá Tân » Châm ngôn mới \(2013\)](#)

Cái làm ta hạnh phúc
Thực ra cũng chẳng nhiều
Chỉ cần có ai đó
Để ta thầm thương yêu

Rồi thêm chút công việc
Cho ta làm hàng ngày
Cuối cùng, chút mơ mộng
Để đưa ta lên mây

Nguồn: *Châm ngôn mới* (thơ), Thái Bá Tân, NXB Lao động, 2013



Poem Data Crawling

❖ Coding Step 5: Take Poem Data

```
43     poem_info = {
44         'title': poem_title,
45         'content': poem_content,
46         'source': poem_src,
47         'url': poem_url
48     }
49
50     datasets.append(poem_info)
51
52     driver.back()
53 except Exception as e:
54     print(e)
55     print(poem_url)
```

1 datasets

```
[{"title": "'Bạn xấu như chiếc bóng'", "content": "Bạn xấu như chiếc bóng\nCứ bám riết theo anh\nKhi anh sáng, rực rỡ\nNhư mặt trời long lanh\nNhưng họ sẽ biến mất\nKhi trời phủ mây đen\nTức là khi anh đổi\nTrong túi không có tiền", "source": "[Thông tin 1 nguồn tham khảo đã được ẩn]", "url": "https://www.thivien.net/Th%C3%A1i-B%C3%A1-T-%C3%A2n/B%E1%BA%A1n-x%E1%BA%A5u-nh%C6%B0-chi%E1%BA%BFc-b%C3%B3ng/poem-4r1MaPThwgzxicR8oy7n0"}, {"title": "'Cái làm ta hạnh phúc'", "content": "Cái làm ta hạnh phúc\nThực ra cũng chẳng nhiều\nChỉ cần có ai đó\nĐể ta thầm thương yêu\n\nRồi thêm chút công việc\nCho ta làm hàng ngày\nCuối cùng, chút mơ mong\nĐể đưa ta lên mây", "source": "[Thông tin 1 nguồn tham khảo đã được ẩn]", "url": "https://www.thivien.net/Th%C3%A1i-B%C3%A1-T-%C3%A2n/C%C3%A1i-l%C3%A1m-ta-h%C3%A1nh-ph%C3%ACc/poem-t24-M5Dn2cxMEDVSrQhtCw"}, {"title": "'Chiều vừa xôp trên tay'", "content": "Chiều vừa xôp trên tay\nChợt nghe thoảng ong bay\nCó ai vừa chết nhỉ\nMây thắt tang trăng gầy\n\nnứt đỗ sao cứ đỗ\nTáo chín cho thật vàng\nEm đẹp cho thêm đẹp\nĐể mắt ngừng lang thang", "source": "[Thông tin 1 nguồn tham khảo đã được ẩn]", "url": "https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA%ADn/Chi%E1%BB%81u-v%E1%BB%ABa-x%E1%BB%91p-tr%C3%AA-n-tay/poem-f31v-s36HNxJNqauIrc7Xw"}, {"title": "'Chơi thân không có nghĩa'", "content": "Chơi thân không có nghĩa\nKhông cãi nhau bao giờ\nKhông làm nhau tức giận\nHay đôi lúc hững hờ\n\nVấn đề là ở chỗ\nBã chơi thân với nhau\nThì phải nên kiềm chế\nĐừng giận dỗi quá lâu", "source": "[Thông tin 1 nguồn tham khảo đã được ẩn]", "url": "https://www.thivien.net/Th%C3%A1i-B%C3%A1-T-%C3%A2n/Ch%C6%A1i-th%C3%A2n-kh%C3%A1ng-c%C3%BCng-ngh%C4%A9a/poem-UHcGs3KT2AX8C0LsaF5T3w"}]
```

Poem Data Crawling

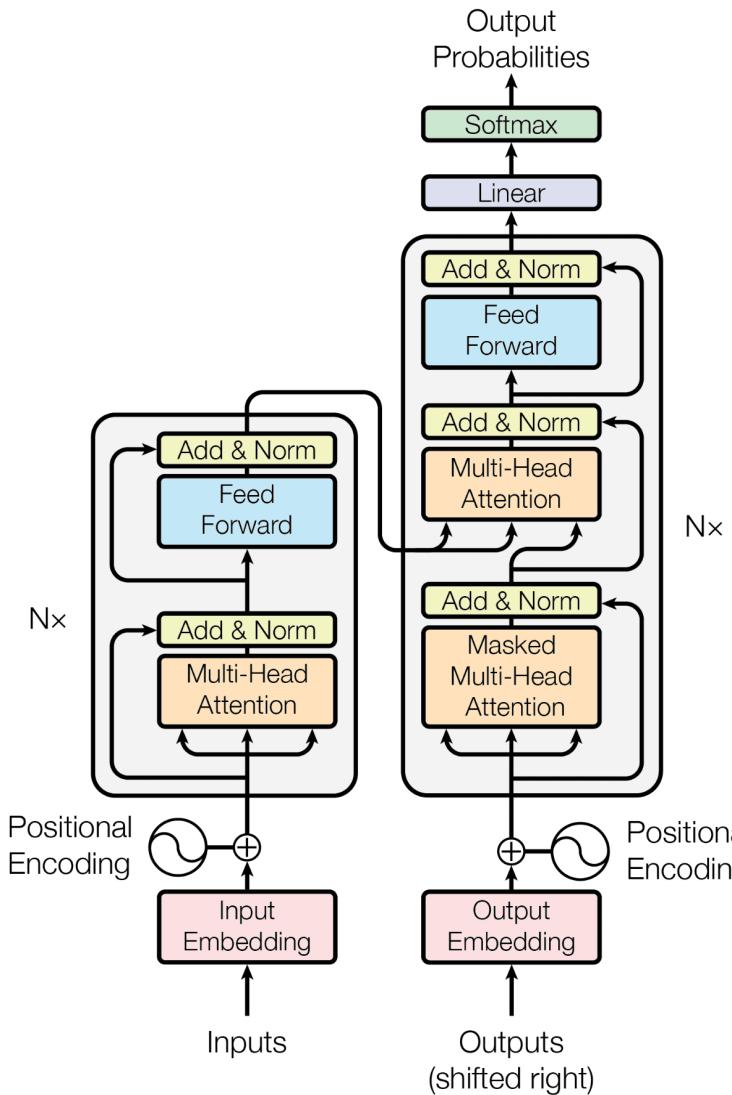
❖ Crawling results

Download link

Poem Generation: From Scratch

Poem Generation: From Scratch

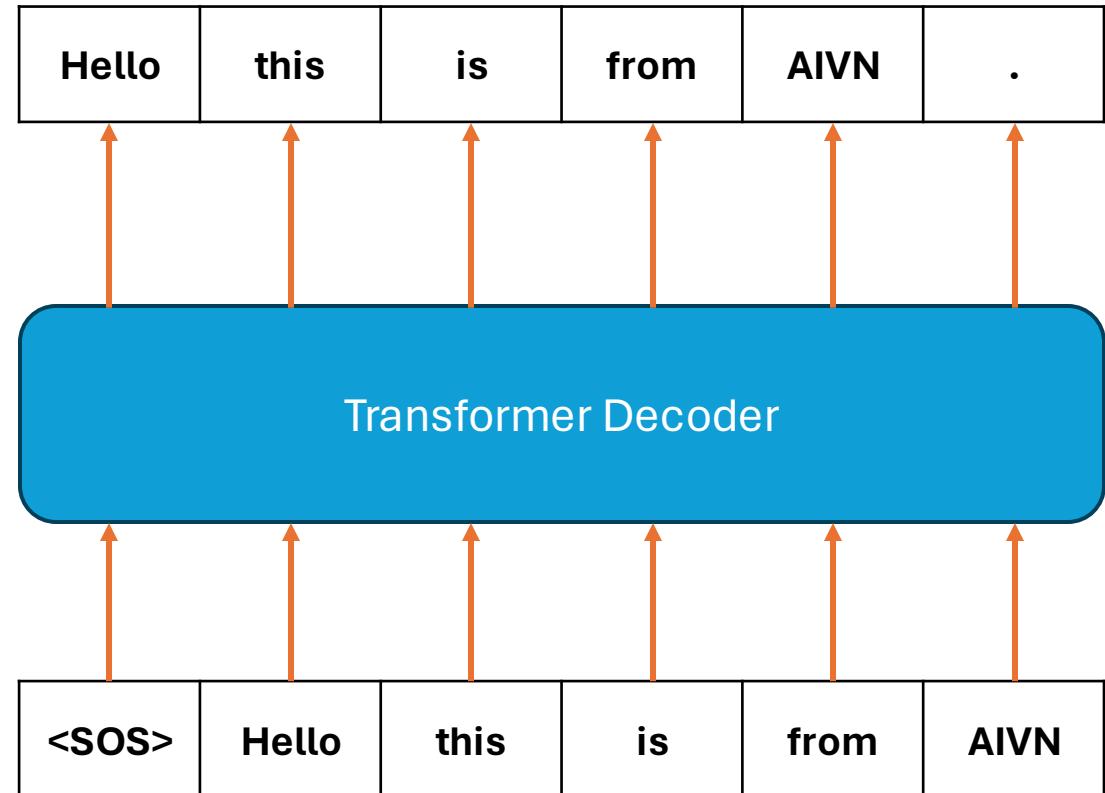
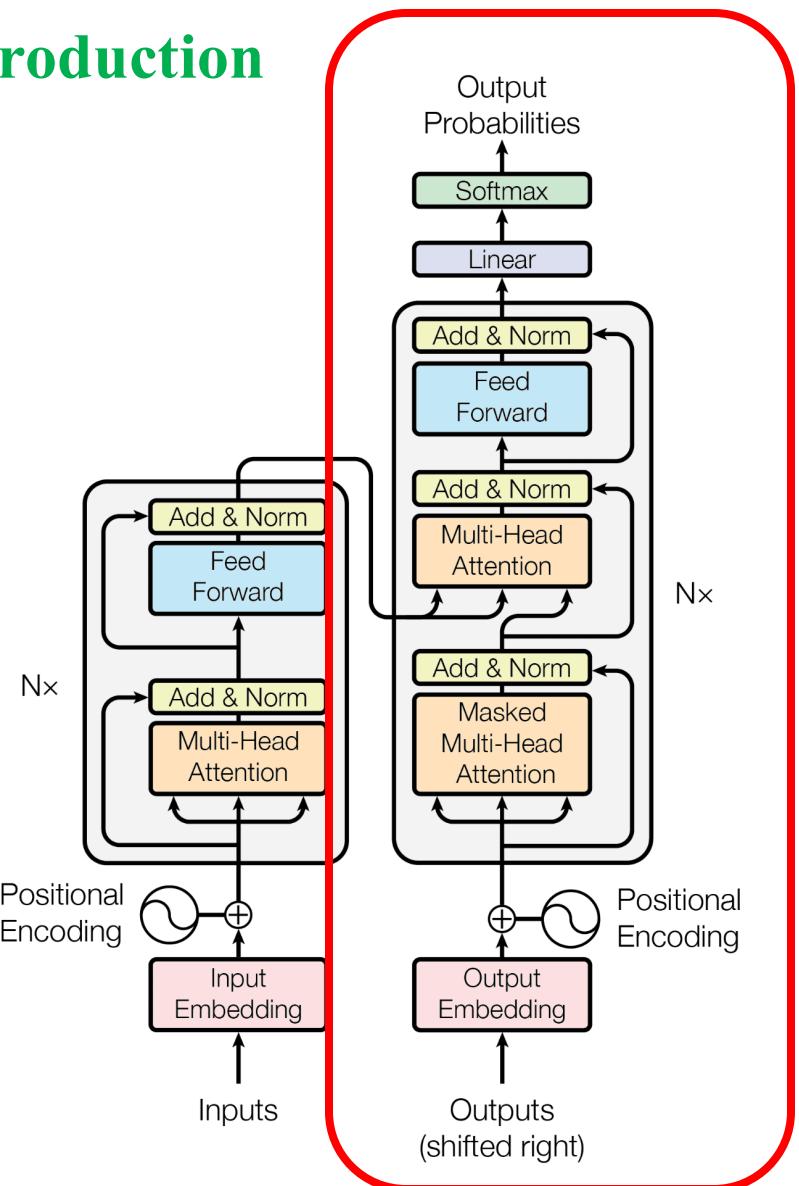
❖ Introduction



Model to be used: **Transformer**

Poem Generation: From Scratch

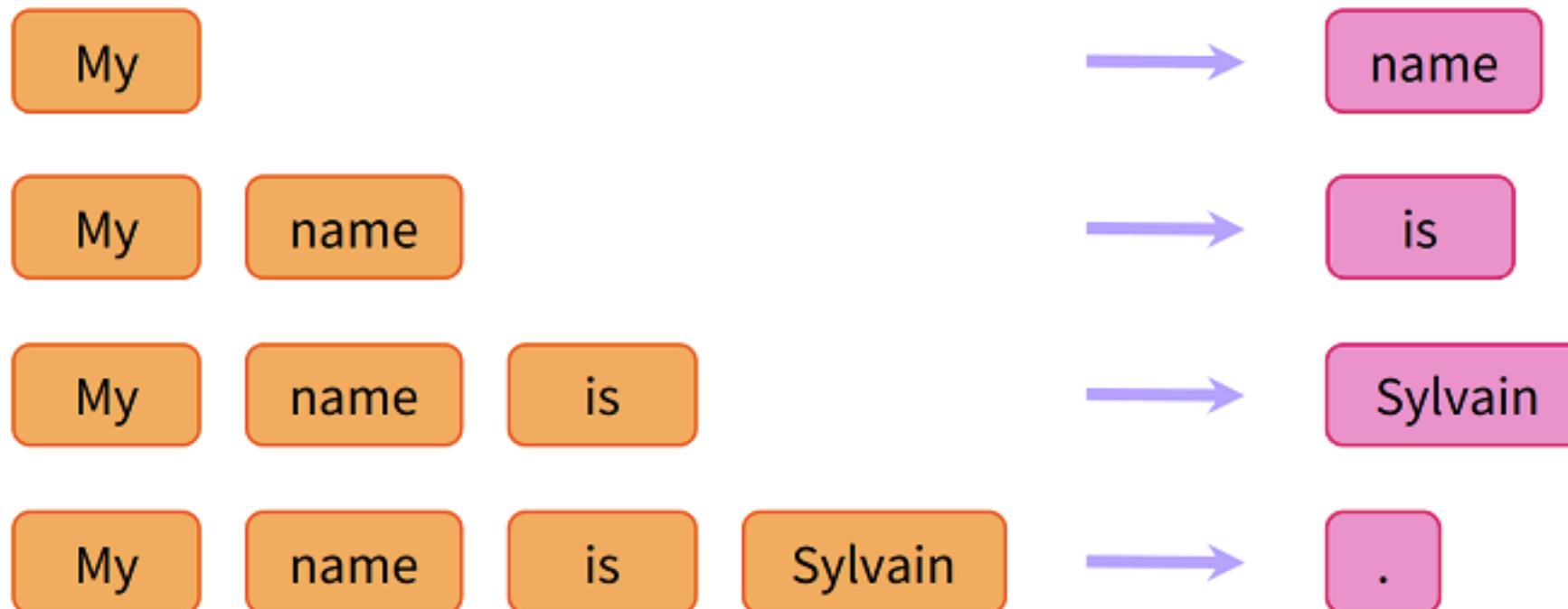
❖ Introduction



Task: Next token prediction (causal language modeling)

Poem Generation: From Scratch

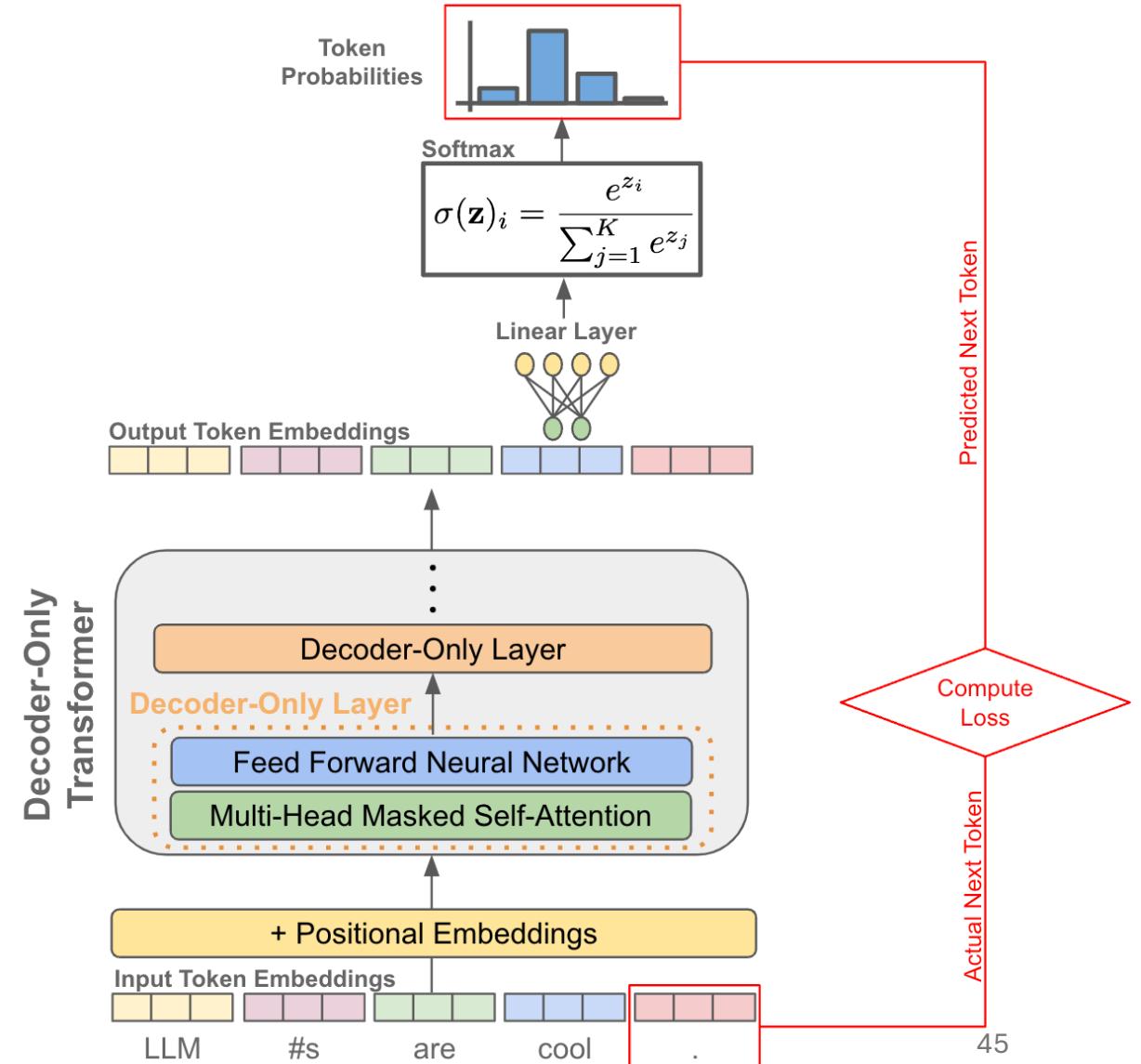
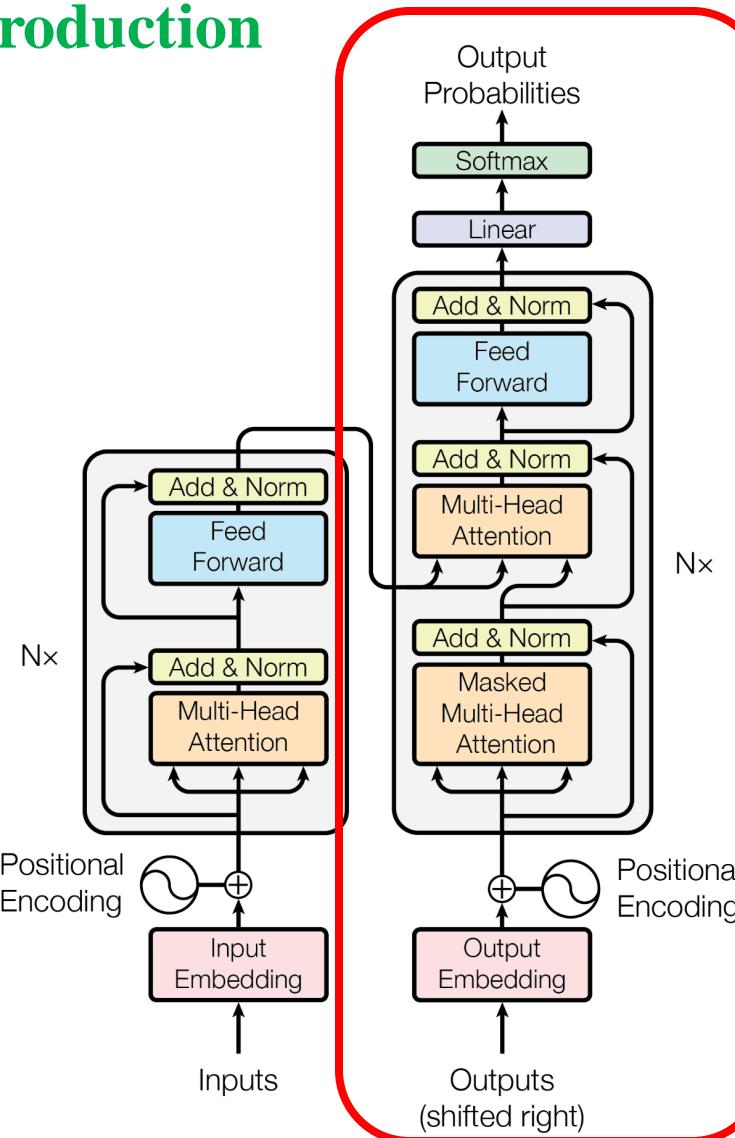
❖ Introduction



Causal Language Modeling (Autoregressive)

Poem Generation: From Scratch

❖ Introduction



Poem Generation: From Scratch

❖ Coding Step 1: Import libraries

```
1 import math
2 import os
3 import re
4 import time
5 import pandas as pd
6 import numpy as np
7
8 import torch
9 import torch.nn as nn
10 import torch.nn.functional as F
11 from torch.utils.data import Dataset, DataLoader
12 from torchtext.data.utils import get_tokenizer
13 from torchtext.vocab import build_vocab_from_iterator
```



Poem Generation: From Scratch

❖ Coding Step 2: Download and read dataset

1 datasets

```
[{"title": '“Bạn xấu như chiếc bóng”',  
 'content': 'Bạn xấu như chiếc bóng\nCứ bám riết theo anh\nKhi anh sáng, rực rỡ\nNhư mặt trời long lanh\nNhưng họ  
sẽ biến mất\nKhi trời phủ mây đen\nTức là khi anh đói\nTrong túi không có tiền',  
 'source': '[Thông tin 1 nguồn tham khảo đã được ẩn]',  
 'url': 'https://www.thivien.net/Th%C3%A1i-B%C3%A1i-T%C3%A2n/B%E1%BA%A1n-x%E1%BA%A5u-nh%C6%B0-chi%E1%BA%BFc-b%C3%B3ng/poem-4r1MApTHwgzgxicR8oy7nQ'},  
 {"title": '“Cái làm ta hạnh phúc”',  
 'content': 'Cái làm ta hạnh phúc\nThực ra cũng chẳng nhiều\nChỉ cần có ai đó\nĐể ta thầm thương yêu\nRồi thêm chút  
công việc\nCho ta làm hàng ngày\nCuối cùng, chút mơ mong\nĐể đưa ta lên mây',  
 'source': '[Thông tin 1 nguồn tham khảo đã được ẩn]',  
 'url': 'https://www.thivien.net/Th%C3%A1i-B%C3%A1i-T%C3%A2n/C%C3%A1i-l%C3%A1m-ta-h%C3%A1nh-ph%C3%ACc-poem-t24-M5Dn2cxmEDVSrQhtCw'},  
 {"title": '“Chiều vừa xôp trên tay”',  
 'content': 'Chiều vừa xôp trên tay\nChợt nghe thoảng ong bay\nCó ai vừa chết nhỉ\nMây thắt tang trăng gầy\nnốt đỏ  
sao cứ đỏ\nTáo chín cho thật vàng\nEm đẹp cho thêm đẹp\nĐể mắt ngừng lang thang',  
 'source': '[Thông tin 1 nguồn tham khảo đã được ẩn]',  
 'url': 'https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA%ADn/Chi%E1%BB%81u-v%E1%BB%ABa-x%E1%BB%91p-tr%C3%AAn-tay/poem-f31v-s36HNxJNqauIrC7Xw'},  
 {"title": '“Chơi thân không có nghĩa”',  
 'content': 'Chơi thân không có nghĩa\nKhông cãi nhau bao giờ\nKhông làm nhau tức giận\nHay đôi lúc hững hờ\nVẫn đè  
là ở chỗ\nĐã chơi thân với nhau\nThì phải nên kiềm chế\nĐừng giận dỗi quá lâu',  
 'source': '[Thông tin 1 nguồn tham khảo đã được ẩn]',  
 'url': 'https://www.thivien.net/Th%C3%A1i-B%C3%A1i-T%C3%A2n/Ch%C6%A1i-th%C3%A2n-kh%C3%B4ng-c%C3%BDng-ngh%C4%A9a/poem-UHcGs3KT2AX8C0LsaFsT3w'},  
 {"title": "“Hai em là hai em”",  
 'content': 'Hai em là hai em\nNhưng không giống nhau\nNhưng không giống nhau\nNhưng không giống nhau',  
 'source': '[Thông tin 1 nguồn tham khảo đã được ẩn]',  
 'url': 'https://www.thivien.net/Th%C3%A1i-B%C3%A1i-T%C3%A2n/Hai-em-la-hai-em/poem-4r1MApTHwgzgxicR8oy7nQ'}]
```

Use what we crawled: [Download link](#)

Poem Generation: From Scratch

❖ Coding Step 2: Download and read dataset

```
1 # https://drive.google.com/file/d/1KfrBAycsgQBt1mtEbzMh5pSYL8YIk0Tc/view?usp=sharing  
2 !gdown --id 1KfrBAycsgQBt1mtEbzMh5pSYL8YIk0Tc  
3 !unzip poem_dataset_final.zip
```

```
1 DATASET_PATH = 'datasets/poem_final.csv'  
2 df = pd.read_csv(DATASET_PATH)  
3 df
```

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...	https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1n-Ch%C3%A1m-ngh%C3%B4n-m%C3%9Bi-%C4%90%C3%AD-th%C3%83o
1	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...	https://www.thivien.net/L%C3%A1m-Huy-Nhu%E1%BA%AFn-Chieu-co-that-tho
2	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	https://www.thivien.net/Nguy%E1%BB%85n-Nh%E1%BA%BFt-Bi%C3%A9c
3	3 “Đến, nhiều nơi để đến”	Đến, nhiều nơi để đến\nVề, trở lại với mình\nC...	Nguồn: Châm ngôn mới (thơ), Thái Bá Tân, NXB L...	https://www.thivien.net/Th%C3%83i-B%C3%A1-T%C3%A1n-Ch%C3%A1m-ngh%C3%B4n-m%C3%9Bi-%C4%90%C3%AD-th%C3%83o
4	4 “Đừng bao giờ dại dột”	Đừng bao giờ dại dột\nĐem chuyện riêng của mìn...	Nguồn: Châm ngôn mới (thơ), Thái Bá Tân, NXB L...	https://www.thivien.net/Th%C3%83i-B%C3%A1-T%C3%A1n-Ch%C3%A1m-ngh%C3%B4n-m%C3%9Bi-%C4%90%C3%AD-th%C3%83o
...

Poem Generation: From Scratch

❖ Coding Step 2: Download and read dataset

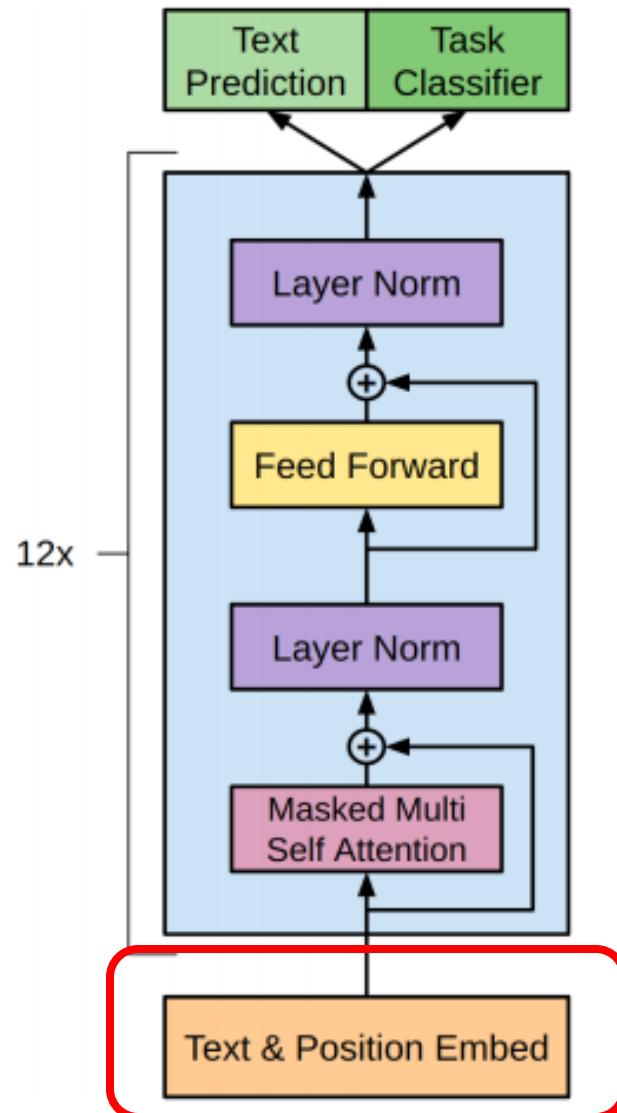
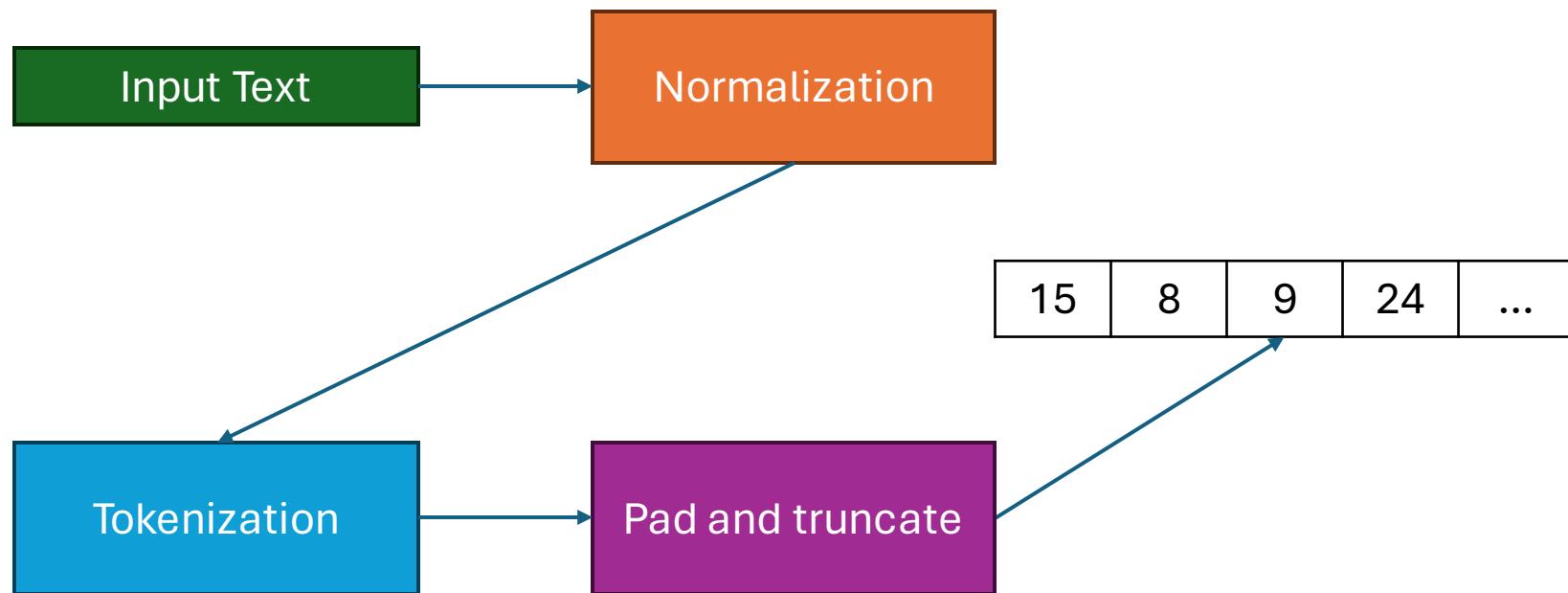
Unnamed:		title	content
0	0	"Cái làm ta hạnh phúc"	Cái làm ta hạnh phúc\nThực ra cũng chẳng nhiều...
1	1	"Chiều vừa xốp trên tay"	Chiều vừa xốp trên tay\nChợt nghe thoảng ong b...
2	2	"Dưới giàn hoa thiên lý..."	Dưới giàn hoa thiên lý\nMột mình anh đang ngồi...
3	3	"Đến, nhiều nơi để đến"	Đến, nhiều nơi để đến\nVề, trở lại với mình\nC...
4	4	"Đừng bao giờ dại dột"	Đừng bao giờ dại dột\nĐem chuyện riêng của mìn...
...

```
1 df['content'][0].split('\n')
```

```
['Cái làm ta hạnh phúc',  
'Thực ra cũng chẳng nhiều',  
'Chỉ cần có ai đó',  
'Để ta thầm thương yêu',  
'',  
'Rồi thêm chút công việc',  
'Cho ta làm hàng ngày',  
'Cuối cùng, chút mơ mộng',  
'Để đưa ta lên mây']
```

Poem Generation: From Scratch

❖ Coding Step 3: Build vectorization function



Poem Generation: From Scratch

❖ Coding Step 3: Build vectorization function

Uppercase
the first
character at
the beginning
of a line.

Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dạy tin yêu.

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lảng lặng,
Trong mơ, ta cùng nhảy.

Use punctuation for
controlling pace/flow,
indicating pause,
emphasizing on
aesthetic or rhythmic
qualities

It appears to be better not to remove punctuation and do lowercasing

Poem Generation: From Scratch

❖ Coding Step 3: Build vectorization function

Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dậy tin yêu.

Đêm thì thăm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lẳng lặng,
Trong mơ, ta cùng nhảy.

```
1 def text_normalize(text):  
2     text = text.strip()  
3     return text  
4  
5  
6 df['content'] = df['content'].apply(lambda x: text_normalize(x))
```

Poem Generation: From Scratch

❖ Coding Step 3: Build vectorization function

Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dậy tin yêu.

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lảng lặng,
Trong mơ, ta cùng nhảy.

```
1 for idx, row in df.iterrows():
2     print(f'{idx+1}.')
3     print(row['content'])
4     print()
5     if idx == 10:
6         break
```

1.
Cái làm ta hạnh phúc
Thực ra cũng chẳng nhiều
Chỉ cần có ai đó
Để ta thầm thương yêu

Rồi thêm chút công việc
Cho ta làm hàng ngày
Cuối cùng, chút mơ mộng
Để đưa ta lên mây

2.
Chiều vừa xôp trên tay
Chợt nghe thoảng ong bay
Có ai vừa chết nhỉ
Mây thắt tang trăng gầy

Ớt đỏ sao cứ đỏ
Táo chín cho thật vàng
Em đẹp cho thêm đẹp
Để mắt ngừng lang thang

Poem Generation: From Scratch

❖ Coding Step 3: Build vectorization function

```
1 def tokenizer(text):
2     return text.split()
3
4 def yield_tokens(df):
5     for idx, row in df.iterrows():
6         yield tokenizer(row['content'])
7
8 vocab = build_vocab_from_iterator(
9     yield_tokens(df),
10    specials=['<unk>', '<pad>', '<sos>', '<eos>', '<eol>']
11 )
12 vocab.set_default_index(vocab['<unk>'])
13 vocab.get_stoi()
```

```
{'Liệu': 2198,
'ảm': 2196,
'âm': 2195,
'dợi,'': 2193,
'dia': 2187,
'dèn!': 2186,
'dêm': 2185,
'dặn': 2184,
'dáng': 2182,
'diên': 2178,
'Dược': 2174,
'Dâu': 2172,
'Dàn': 2171,
'ô-kê': 2169,
'Dức': 2164,
'Dêm': 2162,
'Dã': 2161,
'xấu': 2158,
'xúc': 2156,
'xoãi': 2154,
'veng': 2151,
've...': 2150,
'vặng': 2148,
'tôt': 2144,
'tổng': 2143,
'tất': 2141,
'túi': 2138,
'tôi.': 2137,
```

Poem Generation: From Scratch

❖ Coding Step 3: Build vectorization function

```
1 PAD_TOKEN = vocab['<pad>']
2
3 MAX_SEQ_LEN = 25
4
5 def pad_and_truncate(input_ids, max_seq_len):
6     if len(input_ids) > max_seq_len:
7         input_ids = input_ids[:max_seq_len]
8     else:
9         input_ids += [PAD_TOKEN] * (max_seq_len - len(input_ids))
10
11    return input_ids
12
13 def vectorize(text, max_seq_len):
14     input_ids = [vocab[token] for token in tokenizer(text)]
15     input_ids = pad_and_truncate(input_ids, max_seq_len)
16
17    return input_ids
18
19 def decode(input_ids):
20     return [vocab.get_itos()[token_id] for token_id in input_ids]
```

```
{'Liệu': 2198,
'ảm': 2196,
'âm': 2195,
'dợi,'': 2193,
'dia': 2187,
'dèn!': 2186,
'dêm': 2185,
'dặn': 2184,
'dáng': 2182,
'diên': 2178,
'Dược': 2174,
'Dâu': 2172,
'Dàn': 2171,
'ô-kê': 2169,
'Dức': 2164,
'Dêm': 2162,
'Dã': 2161,
'xấu': 2158,
'xúc': 2156,
'xoãi': 2154,
'veng': 2151,
've...': 2150,
'veng': 2148,
'tôt': 2144,
'tổng': 2143,
'tất': 2141,
'túi': 2138,
'tôi.'': 2137,
```

Poem Generation: From Scratch

❖ Coding Step 3: Build vectorization function

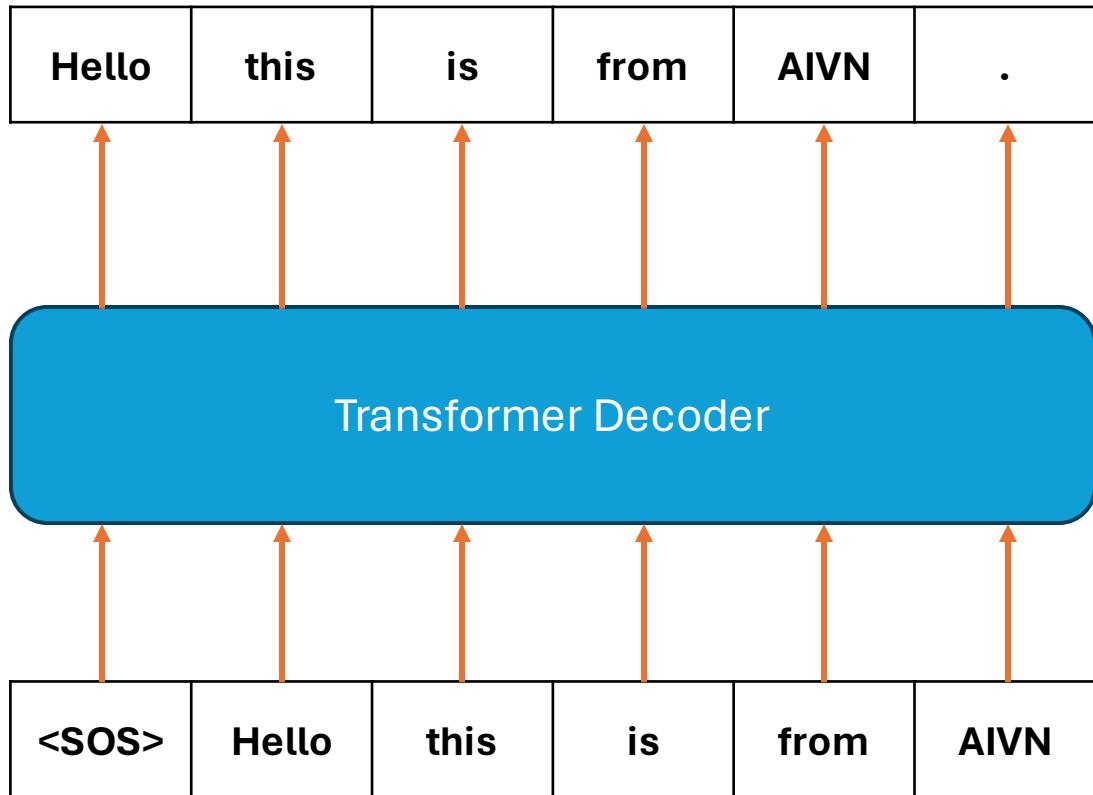
```
1 PAD_TOKEN = vocab['<pad>']
2
3 MAX_SEQ_LEN = 25
4
5 def pad_and_truncate(input_ids, max_seq_len):
6     if len(input_ids) > max_seq_len:
7         input_ids = input_ids[:max_seq_len]
8     else:
9         input_ids += [PAD_TOKEN] * (max_seq_len - len(input_ids))
10
11    return input_ids
12
13 def vectorize(text, max_seq_len):
14     input_ids = [vocab[token] for token in tokenizer(text)]
15     input_ids = pad_and_truncate(input_ids, max_seq_len)
16
17     return input_ids
18
19 def decode(input_ids):
20     return [vocab.get_itos()[token_id] for token_id in input_ids]
```

```
1 print(df['content'][0].split('\n')[0])
2 print(vectorize(df['content'][0].split('\n')[0], 10))

Cái làm ta hạnh phúc
[175, 62, 39, 313, 366, 1, 1, 1, 1, 1]
```

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset



Input: “<SOS> Hello this is from AIVN”

Output: “Hello this is from AIVN .”

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dạy tin yêu.

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lẳng lặng,
Trong mơ, ta cùng nhảy.

0	1	2	3	4
Hạnh	phúc	trong	điều	giản,

Iterate through each word:

Sample 1:

Input: "<SOS>"
Output: "Hạnh"

Sample 2:

Input: "<SOS> Hạnh"
Output: "Hạnh phúc"

Sample 3:

Input: "<SOS> Hạnh phúc"
Output: "Hạnh phúc trong"

Sample 4:

Input: "<SOS> Hạnh phúc trong"
Output: "Hạnh phúc trong điều"

Sample 5:

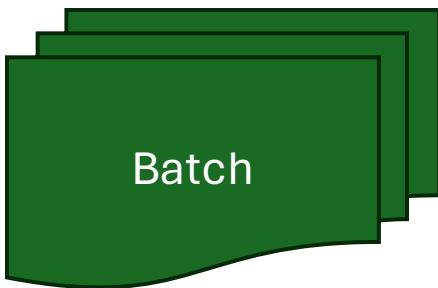
Input: "<SOS> Hạnh phúc trong điều"
Output: "Hạnh phúc trong điều giản,"

Sample 6:

Input: "<SOS> Hạnh phúc trong điều giản,"
Output: "Hạnh phúc trong điều giản <EOS>"

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset



All samples must be of the same length for batching

Sample 1:

Input: "<SOS>"

Output: "Hạnh"

Sample 2:

Input: "<SOS> Hạnh"

Output: "Hạnh phúc"

Sample 3:

Input: "<SOS> Hạnh phúc"

Output: "Hạnh phúc trong"

**Sample 1:**

Input: "<SOS> <PAD> <PAD> <PAD>"

Output: "Hạnh <PAD> <PAD> <PAD>"

Sample 2:

Input: "<SOS> Hạnh <PAD> <PAD>"

Output: "Hạnh phúc <PAD> <PAD>"

Sample 3:

Input: "<SOS> Hạnh phúc <PAD>"

Output: "Hạnh phúc trong <PAD>"

E.g: max_seq_len = 4

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dạy tin yêu.

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lặng lặng,
Trong mơ, ta cùng nhảy.



Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dạy tin yêu.

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lặng lặng,
Trong mơ, ta cùng nhảy.

Our generating objective: Generate 4 lines

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dạy tin yêu.

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lẳng lặng,
Trong mơ, ta cùng nhảy.



Our generating objective: Generate 4 lines

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dạy tin yêu.

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lẳng lặng,
Trong mơ, ta cùng nhảy.

How to help model learn:

1. Where to make a new line?
2. Where is the start of the poem?
3. Where is the end of the poem?

Our generating objective: Generate 4 lines

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dạy tin yêu.

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lồng lồng,
Trong mơ, ta cùng nhảy.

MAX_SEQ_LEN = 26

<SOS> Hạnh phúc trong điều giản, <EOL>
Ai kia, lòng khẽ hát. <EOL>
Công việc hàng ngày vui, <EOL>
Ước mơ dạy tin yêu. <EOL> <EOS>

- ❖ **<SOS>**: The beginning of the poem.
- ❖ **<EOS>**: The end of the poem.
- ❖ **<EOL>**: Linebreak.

Our generating objective: Generate 4 lines

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

```
[ '<sos>', 'Cái', 'làm', 'ta', 'hạnh', 'phúc', '<eol>', 'Thực', '<pad>', '<pad>',  
'<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>',  
'<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>' ]
```



```
[ 'Cái', 'làm', 'ta', 'hạnh', 'phúc', '<eol>', 'Thực', 'ra', '<pad>', '<pad>',  
'<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>',  
'<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>' ]
```

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

```
1 class PoemDataset(Dataset):
2     def __init__(self, df, tokenizer, vectorizer, max_seq_len):
3         self.tokenizer = tokenizer
4         self.vectorizer = vectorizer
5         self.max_seq_len = max_seq_len
6         self.input_seqs, self.target_seqs, self.padding_masks = self.create_samples(df)
```

PoemDataset Implementation

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

```
63     def __len__(self):  
64         return len(self.input_seqs)  
65  
66     def __getitem__(self, idx):  
67         input_seqs = self.input_seqs[idx]  
68         target_seqs = self.target_seqs[idx]  
69         padding_masks = self.padding_masks[idx]  
70  
71     return input_seqs, target_seqs, padding_masks
```

} Build functions to extract these

PoemDataset Implementation

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dạy tin yêu.

Split

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lẳng lặng,
Trong mơ, ta cùng nhảy.

```
def split_content(self, content):
    samples = []

    poem_parts = content.split('\n\n')
    for poem_part in poem_parts:
        poem_in_lines = poem_part.split('\n')
        if len(poem_in_lines) == 4:
            samples.append(poem_in_lines)

    return samples
```

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

```
def create_samples(self, df):
    input_seqs = []
    target_words = []
    padding_masks = []

    for idx, row in df.iterrows():
        content = row['content']
        samples = self.split_content(content)  Split into parts
        for sample in samples:
            sample_input_seqs, sample_target_words, sample_padding_masks = self.prepare_sample(sample)

            input_seqs += sample_input_seqs
            target_words += sample_target_words
            padding_masks += sample_padding_masks

    input_seqs = torch.tensor(input_seqs, dtype=torch.long)
    target_words = torch.tensor(target_words, dtype=torch.long)
    padding_masks = torch.tensor(padding_masks, dtype=torch.float)

    return input_seqs, target_words, padding_masks
```

Create training samples from each part

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

```
def prepare_sample(self, sample):
    input_seqs = []
    target_seqs = []
    padding_masks = []

    input_text = '<sos> ' + ' <eol> '.join(sample) + ' <eol> <eos>'
    input_ids = self.tokenizer(input_text)
    for idx in range(1, len(input_ids)):
        input_seq = ' '.join(input_ids[:idx])
        target_seq = ' '.join(input_ids[1:idx+1])
        input_seq = self.vectorizer(input_seq, self.max_seq_len)
        target_seq = self.vectorizer(target_seq, self.max_seq_len)
        padding_mask = self.create_padding_mask(input_seq)

        input_seqs.append(input_seq)
        target_seqs.append(target_seq)
        padding_masks.append(padding_mask)

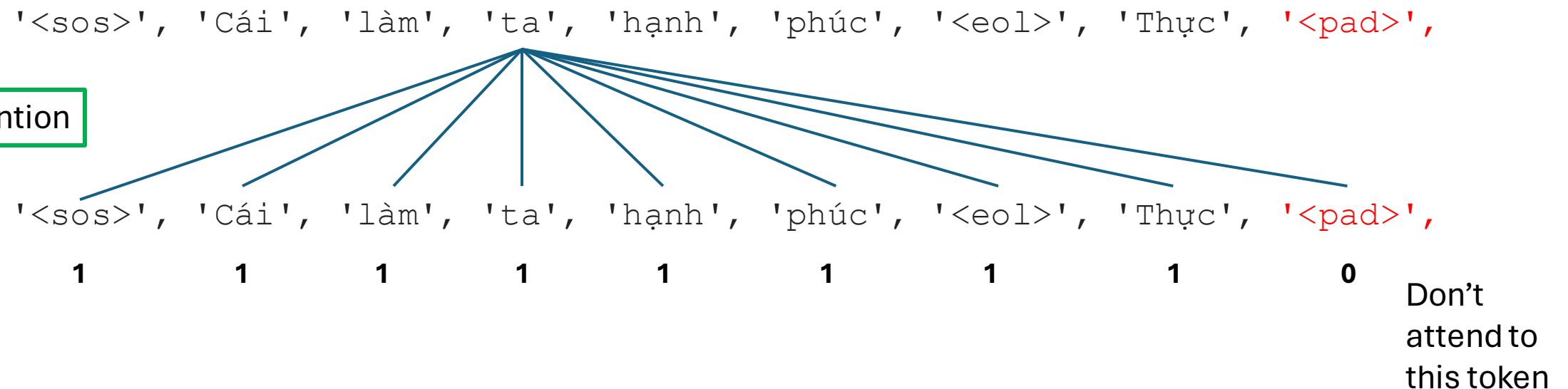
    return input_seqs, target_seqs, padding_masks
```

<SOS> Hạnh phúc trong điều giản, <EOL>
Ai kia, lòng khẽ hát. <EOL>
Công việc hàng ngày vui, <EOL>
Ước mơ dại tin yêu. <EOL> <EOS>

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

```
[ '<sos>', 'Cái', 'làm', 'ta', 'hạnh', 'phúc', '<eol>', 'Thực', '<pad>', '<pad>',  
'<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>',  
'<pad>', '<pad>', '<pad>', '<pad>', '<pad>', '<pad>' ]
```



Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

'<sos>', 'Cái', 'làm', 'ta', 'hạnh', 'phúc', '<eol>', 'Thực', '<pad>>,

Self-attention

'<sos>', 'Cái', 'làm', 'ta', 'hạnh', 'phúc', '<eol>', 'Thực', '<pad>>,

1

1

1

1

1

1

1

1

0

Don't
attend to
this token

```
def create_padding_mask(self, input_ids, pad_token_id=PAD_TOKEN):  
    return [0 if token_id == pad_token_id else 1 for token_id in input_ids]
```

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

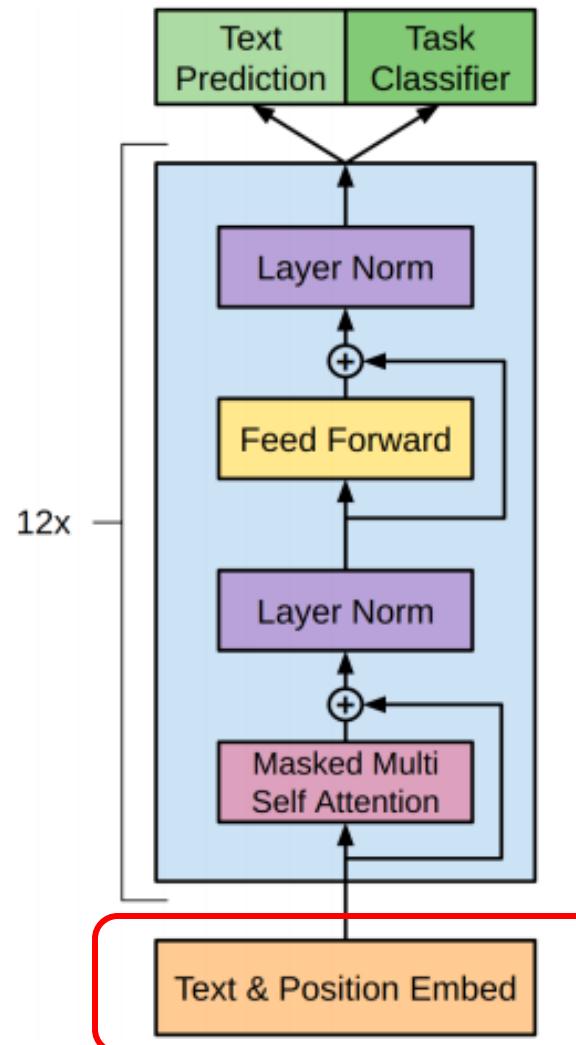
```
73 TRAIN_BS = 256
74 train_dataset = PoemDataset(
75     df=df,
76     tokenizer=tokenizer,
77     vectorizer=vectorize,
78     max_seq_len=MAX_SEQ_LEN
79 )
80 train_loader = DataLoader(
81     train_dataset,
82     batch_size=TRAIN_BS,
83     shuffle=False
84 )
```

Poem Generation: From Scratch

❖ Coding Step 4: Create pytorch dataset

Poem Generation: From Scratch

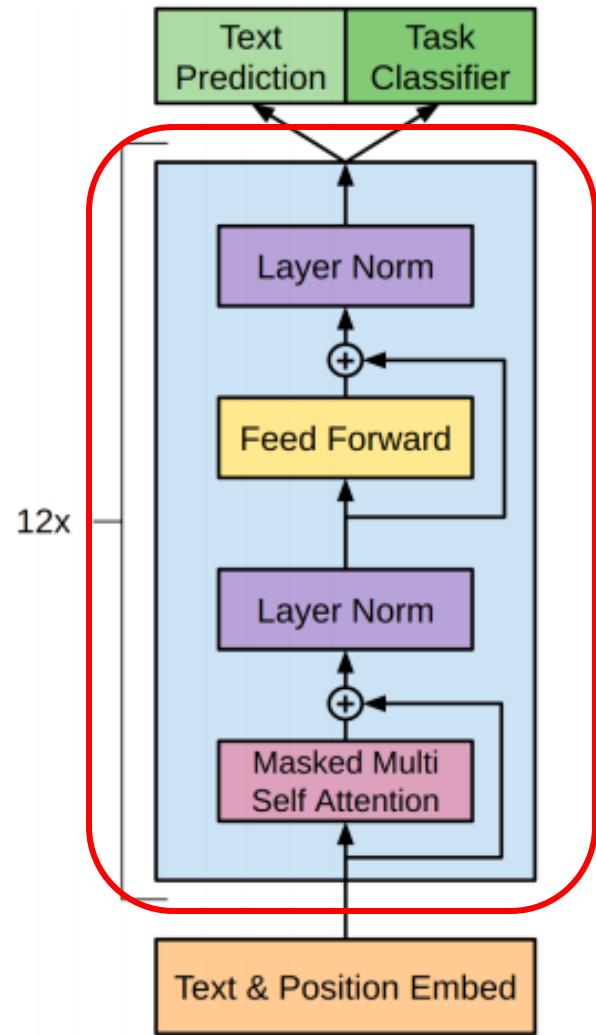
❖ Coding Step 5: Create model



```
1 class PositionalEncoding(nn.Module):
2     def __init__(self, embedding_dims, dropout=0.1, max_len=5000):
3         super(PositionalEncoding, self).__init__()
4         self.dropout = nn.Dropout(p=dropout)
5
6         position = torch.arange(max_len).unsqueeze(1)
7         div_term = torch.exp(torch.arange(0, embedding_dims, 2) * (-math.log(10000.0) / embedding_dims))
8         pe = torch.zeros(max_len, 1, embedding_dims)
9         pe[:, 0, 0::2] = torch.sin(position * div_term)
10        pe[:, 0, 1::2] = torch.cos(position * div_term)
11        self.register_buffer('pe', pe)
12
13    def forward(self, x):
14        x = x + self.pe[:x.size(0)]
15        x = self.dropout(x)
16
17        return x
```

Poem Generation: From Scratch

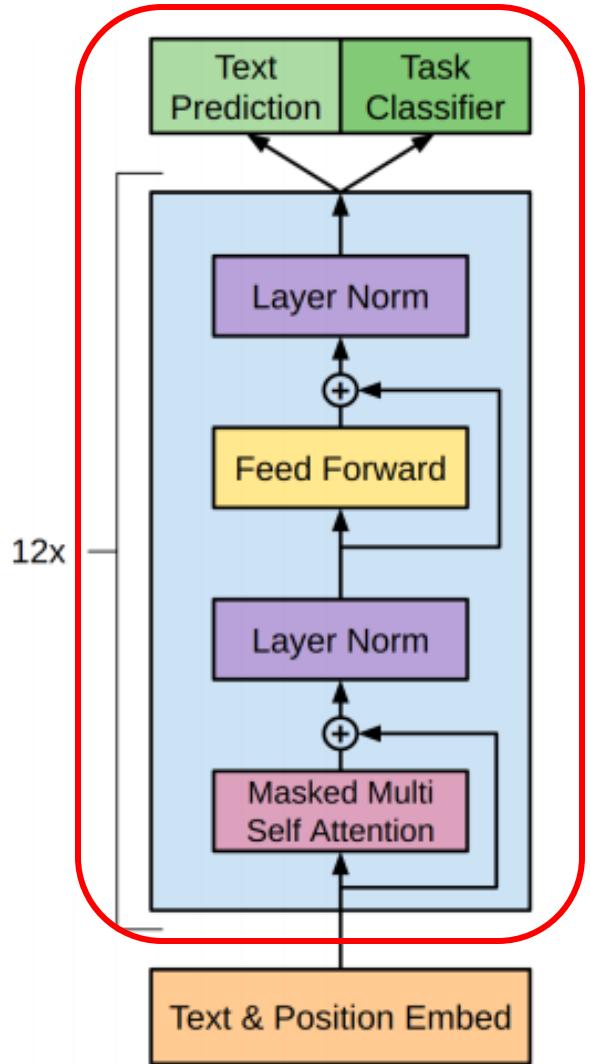
❖ Coding Step 5: Create model



```
19 class TransformerModel(nn.Module):
20     def __init__(
21         self,
22         vocab_size,
23         embedding_dims,
24         n_heads,
25         hidden_dims,
26         n_layers,
27         dropout=0.5
28     ):
29         super(TransformerModel, self).__init__()
30         self.model_type = 'Transformer'
31         self.embedding = nn.Embedding(vocab_size, embedding_dims)
32         self.embedding_dims = embedding_dims
33
34         self.pos_encoder = PositionalEncoding(embedding_dims, dropout)
35         encoder_layers = nn.TransformerEncoderLayer(
36             embedding_dims,
37             n_heads,
38             hidden_dims,
39             dropout
40         )
41         self.transformer_encoder = nn.TransformerEncoder(encoder_layers, n_layers)
42         self.linear = nn.Linear(embedding_dims, vocab_size)
43
44         self.init_weights()
```

Poem Generation: From Scratch

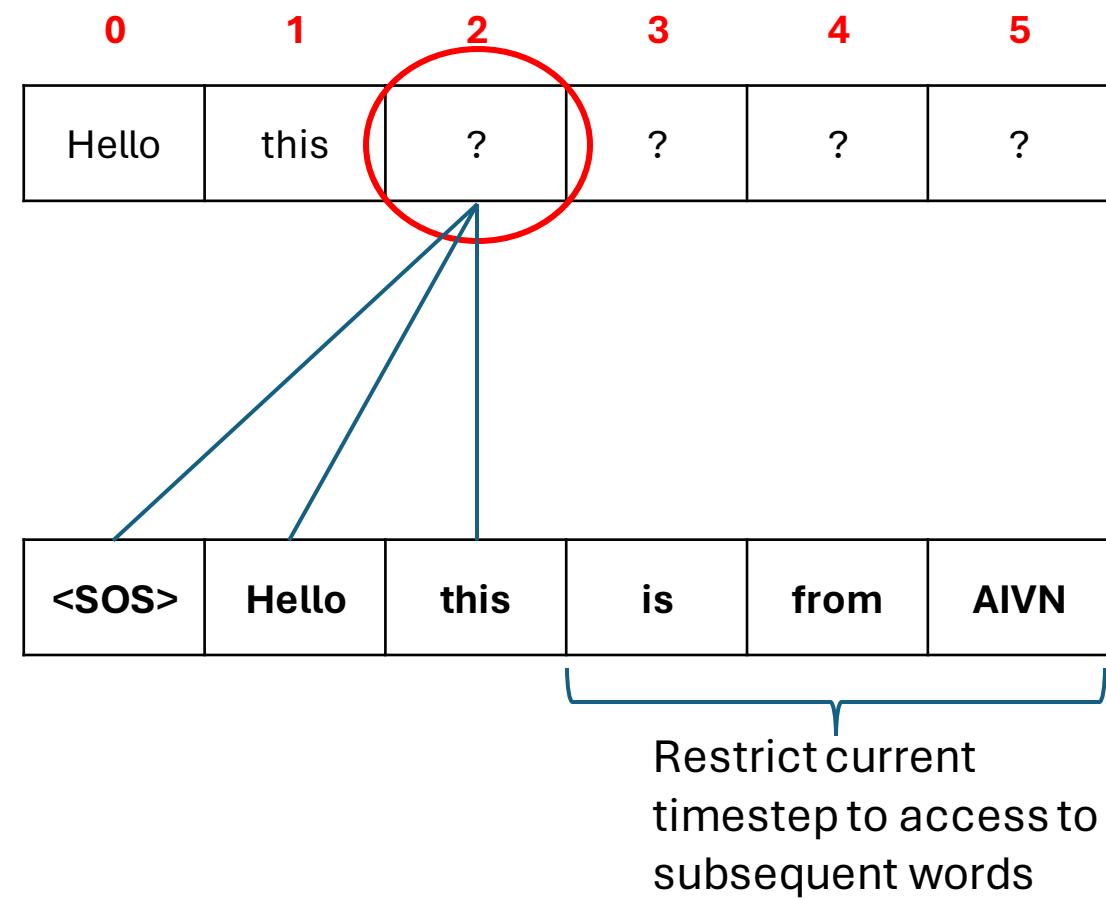
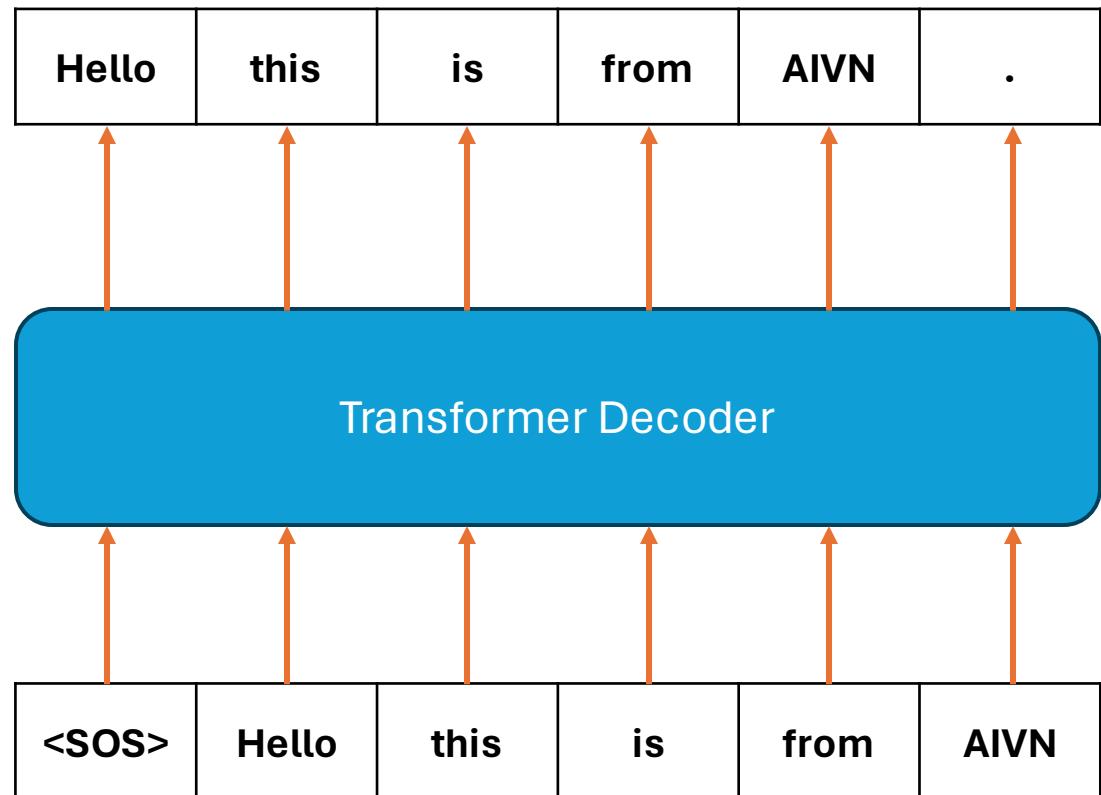
❖ Coding Step 5: Create model



```
46 def init_weights(self):
47     initrange = 0.1
48     self.embedding.weight.data.uniform_(-initrange, initrange)
49     self.linear.bias.data.zero_()
50     self.linear.weight.data.uniform_(-initrange, initrange)
51
52 def forward(self, src, src_mask=None, padding_mask=None):
53     src = self.embedding(src) * math.sqrt(self.embedding_dims)
54     src = self.pos_encoder(src)
55     if src_mask is None:
56         src_mask = nn.Transformer.generate_square_subsequent_mask(len(src)).to(device)
57     output = self.transformer_encoder(src, mask=src_mask, src_key_padding_mask=padding_mask)
58     output = self.linear(output)
59
60     return output
```

Poem Generation: From Scratch

❖ Coding Step 5: Create model



Poem Generation: From Scratch

❖ Coding Step 5: Create model

GPT2: Causal Mask

OUTPUT

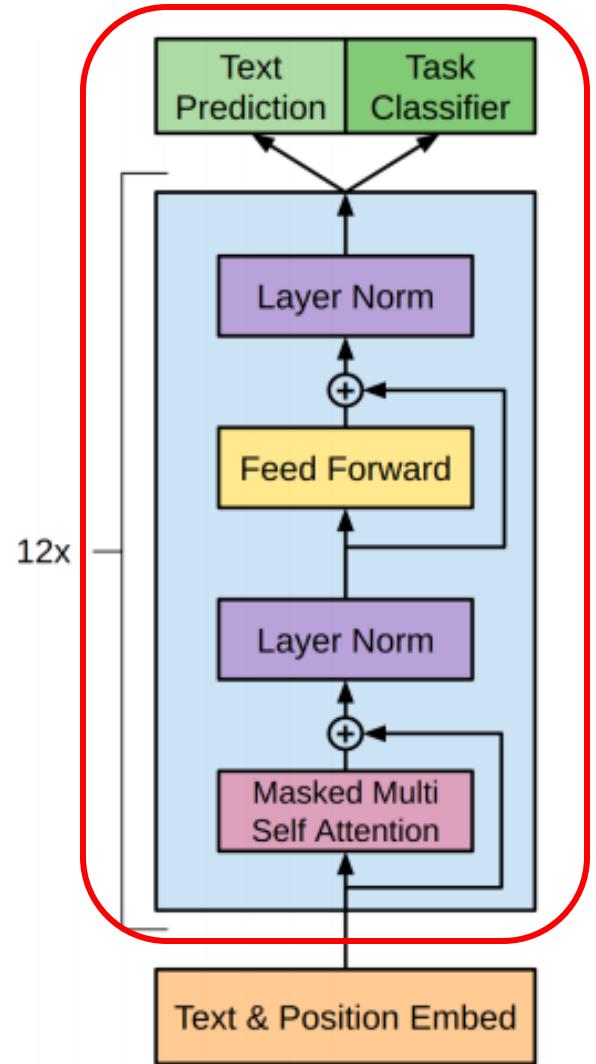
<EOS>	1	1	1	1	1
lunch	1	1	1	1	0
eating	1	1	1	0	0
love	1	1	0	0	0
I	1	0	0	0	0
INPUT =>	<BOS>	-	love	eating	lunch

```
1 src_mask = nn.Transformer.generate_square_subsequent_mask(10)
2 src_mask

tensor([[0., -inf, -inf, -inf, -inf, -inf, -inf, -inf, -inf],
        [0., 0., -inf, -inf, -inf, -inf, -inf, -inf, -inf],
        [0., 0., 0., -inf, -inf, -inf, -inf, -inf, -inf],
        [0., 0., 0., 0., -inf, -inf, -inf, -inf, -inf],
        [0., 0., 0., 0., 0., -inf, -inf, -inf, -inf],
        [0., 0., 0., 0., 0., 0., -inf, -inf, -inf],
        [0., 0., 0., 0., 0., 0., 0., -inf, -inf],
        [0., 0., 0., 0., 0., 0., 0., 0., -inf],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., -inf],
        [0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]])
```

Poem Generation: From Scratch

❖ Coding Step 5: Create model



```
1 VOCAB_SIZE = len(vocab)
2 EMBEDDING_DIMS = 128
3 HIDDEN_DIMS = 128
4 N_LAYERS = 2
5 N_HEADS = 4
6 DROPOUT = 0.2
7
8 device = 'cuda' if torch.cuda.is_available() else 'cpu'
9 input_tests = torch.randint(1, 10, (1, 10)).to(device)
10
11 model = TransformerModel(
12     VOCAB_SIZE,
13     EMBEDDING_DIMS,
14     N_HEADS,
15     HIDDEN_DIMS,
16     N_LAYERS,
17     DROPOUT
18 ).to(device)
19
20 with torch.no_grad():
21     output = model(input_tests)
22     print(output.shape)
```

torch.Size([1, 10, 2201])

Poem Generation: From Scratch

❖ Coding Step 6: Training

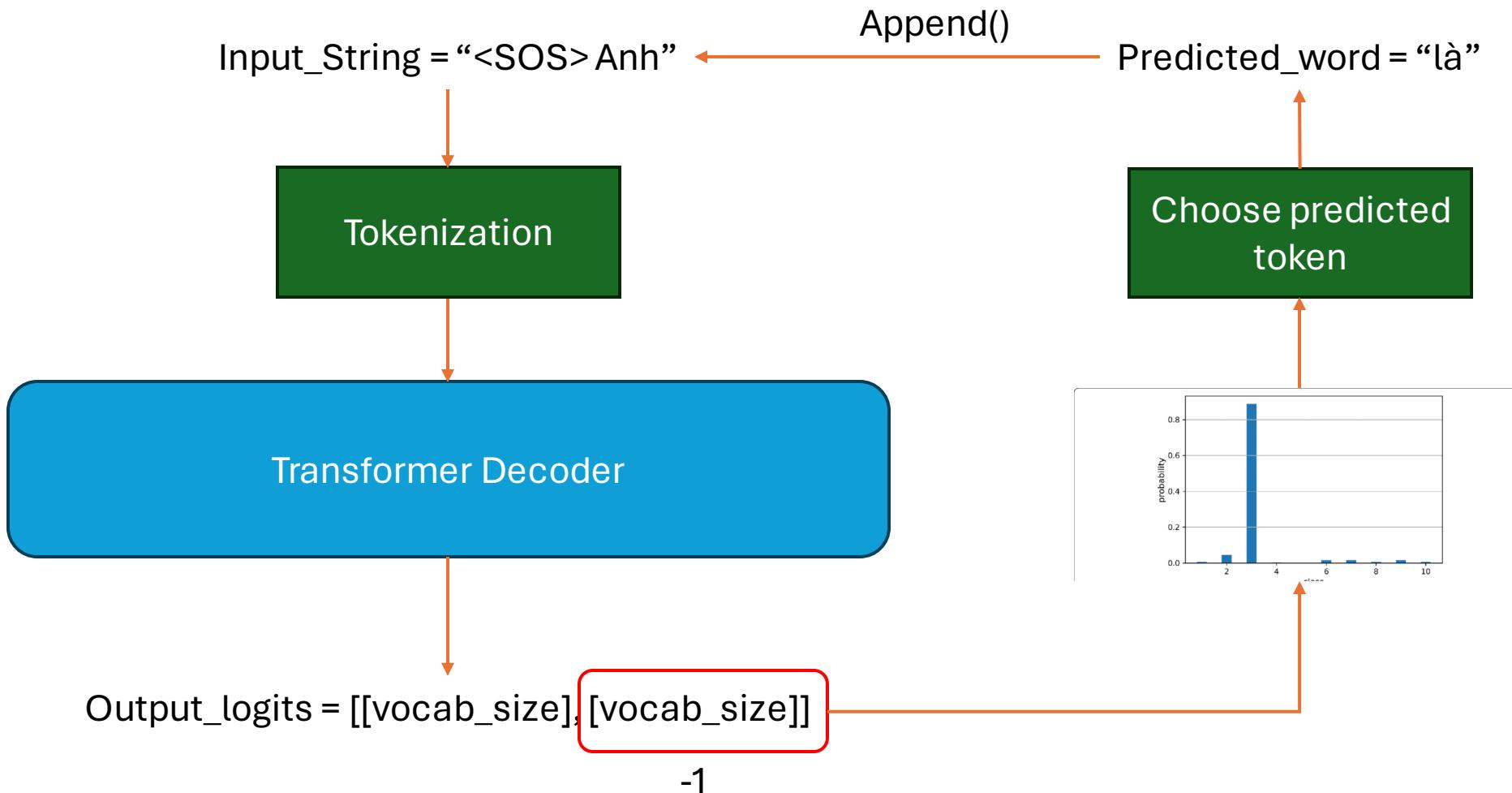
```
1 LR = 5.0
2 EPOCHS = 100
3
4 criterion = nn.CrossEntropyLoss()
5 optimizer = torch.optim.SGD(model.parameters(), lr=LR)
6 scheduler = torch.optim.lr_scheduler.StepLR(optimizer, 1, gamma=0.95)

1 model.train()
2 for epoch in range(EPOCHS):
3     losses = []
4     for idx, samples in enumerate(train_loader):
5         input_seqs, target_seqs, padding_masks = samples
6         input_seqs = input_seqs.to(device)
7         target_seqs = target_seqs.to(device)
8         padding_masks = padding_masks.to(device).permute(1, 0)
9
10        output = model(input_seqs, padding_mask=padding_masks)
11        output = output.permute(0, 2, 1)
12        loss = criterion(output, target_seqs)
13
14        optimizer.zero_grad()
15        loss.backward()
16        torch.nn.utils.clip_grad_norm_(model.parameters(), 0.5)
17        optimizer.step()
18
19        losses.append(loss.item())
20
21    total_loss = sum(losses) / len(losses)
22    print(f'EPOCH {epoch+1}\tLoss {total_loss}')
23    scheduler.step()
```

EPOCH 1	Loss 4.639690832658247
EPOCH 2	Loss 3.625998681241816
EPOCH 3	Loss 3.2110321792689236
EPOCH 4	Loss 2.788922759619626
EPOCH 5	Loss 2.283224804834886
EPOCH 6	Loss 1.8168869641694156
EPOCH 7	Loss 1.4792465662414378
EPOCH 8	Loss 1.2354302108287811
EPOCH 9	Loss 1.0571322508833625
EPOCH 10	Loss 0.9457327005538073
EPOCH 11	Loss 0.8403109799731862
EPOCH 12	Loss 0.7786144885149869
EPOCH 13	Loss 0.7101930718530308
EPOCH 14	Loss 0.691323733465238
EPOCH 15	Loss 0.6552018868652257
EPOCH 16	Loss 0.6382090970873833
EPOCH 17	Loss 0.6235277822071855
EPOCH 18	Loss 0.6107619580897418
EPOCH 19	Loss 0.5927509387785738
EPOCH 20	Loss 0.5788861607963388
EPOCH 21	Loss 0.5693113173950802
EPOCH 22	Loss 0.5591132322495634
EPOCH 23	Loss 0.5465741496194493
EPOCH 24	Loss 0.5366708399219946
EPOCH 25	Loss 0.528262811628255
EPOCH 26	Loss 0.5211723704229702
EPOCH 27	Loss 0.5142657607793808
EPOCH 28	Loss 0.5065084004944022
EPOCH 29	Loss 0.5007355809211731
EPOCH 30	Loss 0.4971333308653398

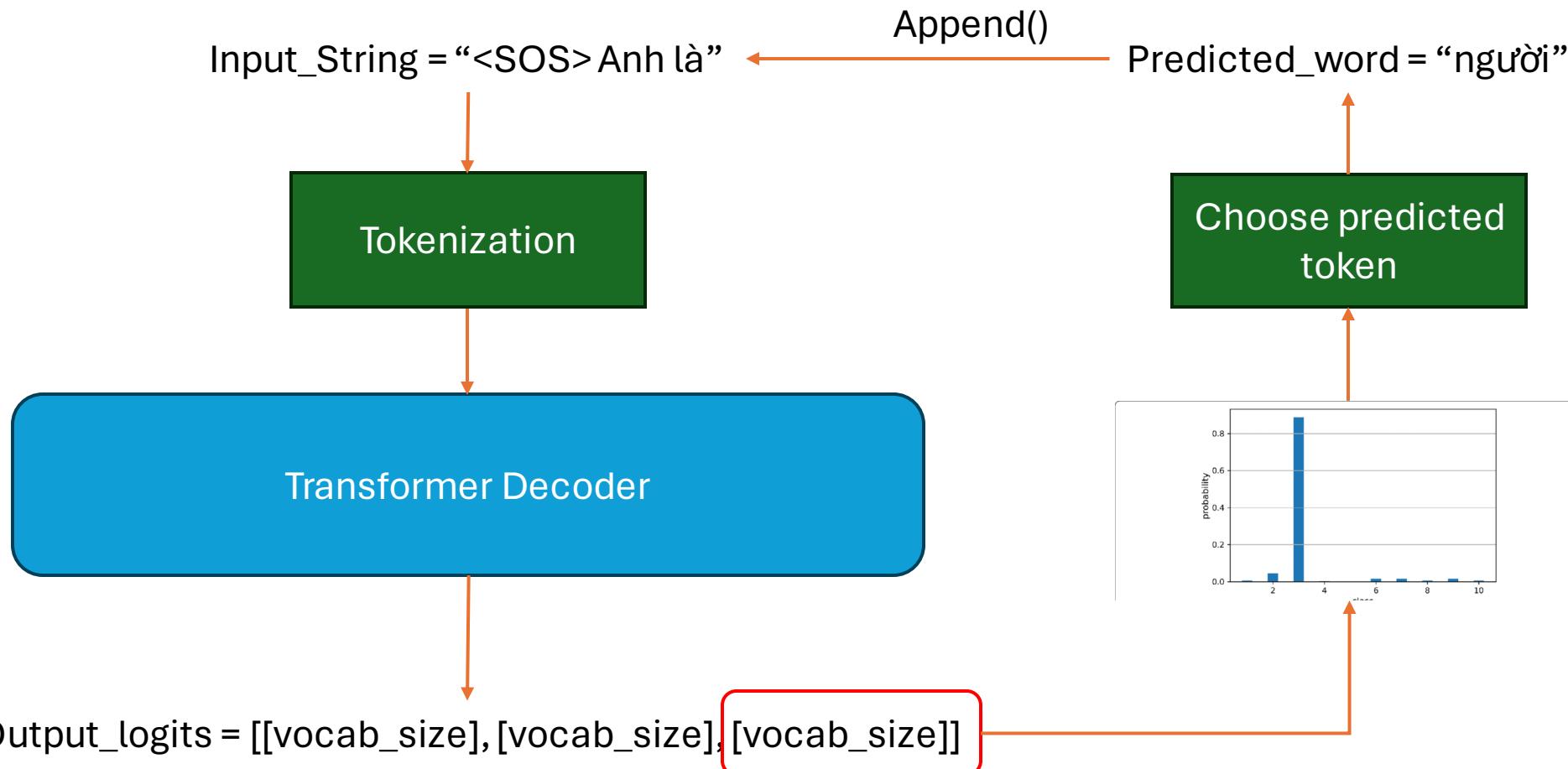
Poem Generation: From Scratch

❖ Coding Step 7: Inference



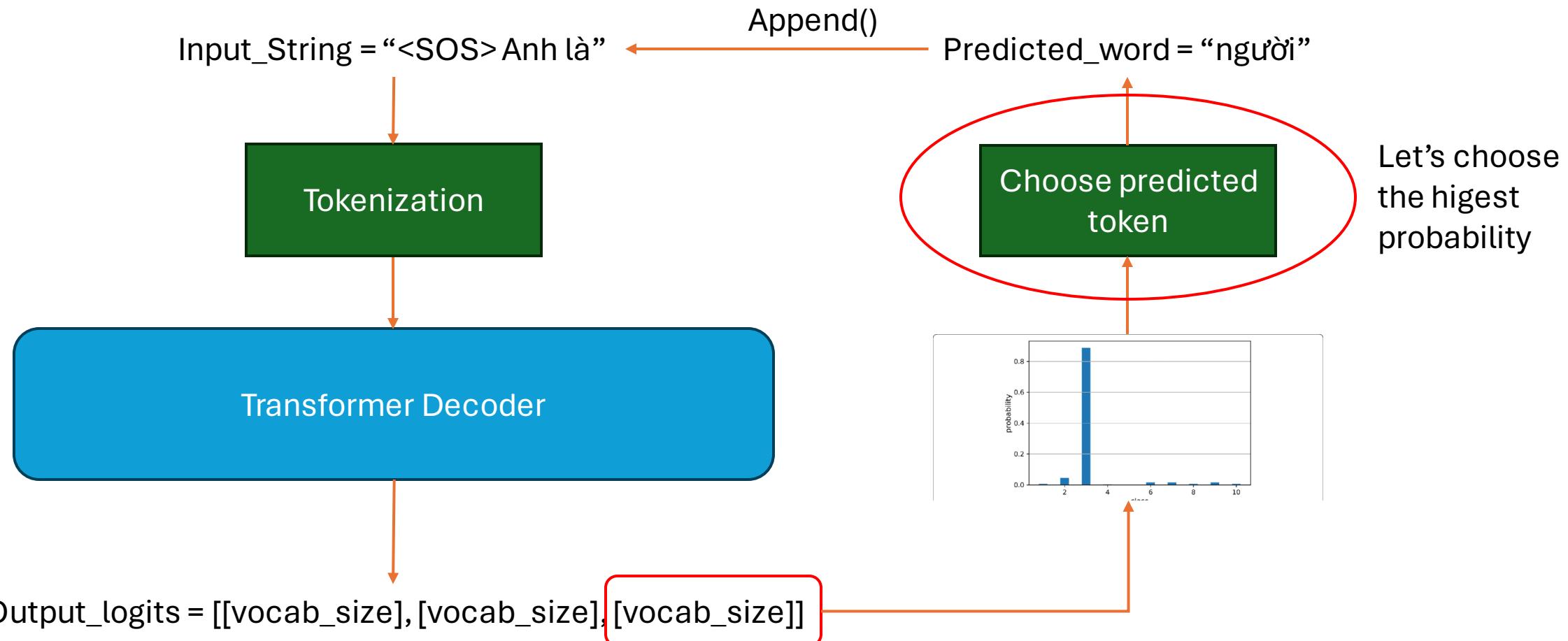
Poem Generation: From Scratch

❖ Coding Step 7: Inference



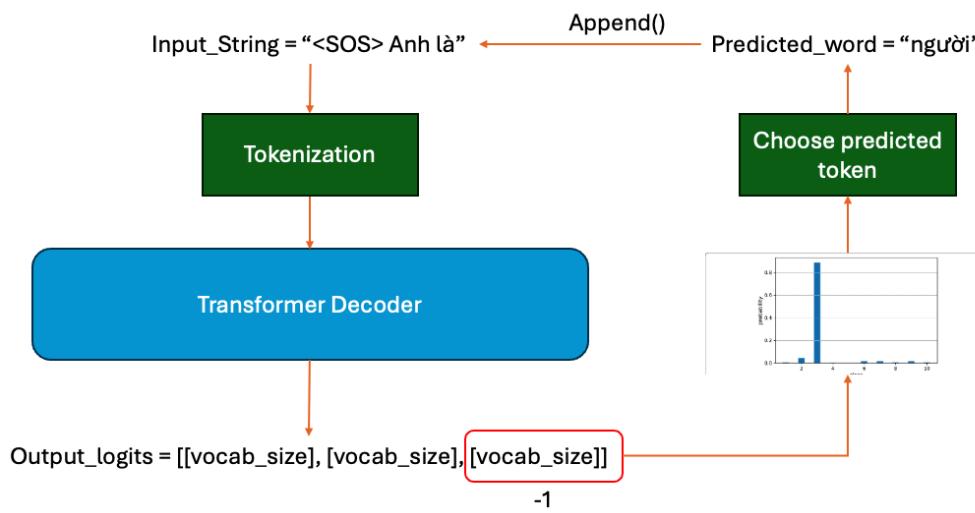
Poem Generation: From Scratch

❖ Coding Step 7: Inference



Poem Generation: From Scratch

❖ Coding Step 7: Inference



```
1 model.eval()
2 input_text = '<sos> Anh'
3 input_tokens = tokenizer(input_text)
4 input_ids = [vocab[token] for token in input_tokens]
5 eos_token_id = vocab['<eos>']
6 generated_ids = input_ids.copy()
7 MAX_GENERATION_LEN = 50
8
9 for _ in range(MAX_GENERATION_LEN):
10     input_tensor = torch.tensor([generated_ids], dtype=torch.long).to(device)
11     with torch.no_grad():
12         outputs = model(input_tensor)
13
14     last_token_logits = outputs[0, -1, :]
15     next_token_id = last_token_logits.argmax().item()
16     generated_ids.append(next_token_id)
17
18     if next_token_id == eos_token_id:
19         break
20
21 # Convert the generated tokens back to text
22 generated_text = decode(generated_ids)
23 generated_text = ' '.join(generated_text)
24 generated_text = generated_text.replace('<sos>', '')
25 lines = generated_text.split('<eol>')
26 for line in lines:
27     print(''.join(line))
```

❖ Coding Step 7: Inference

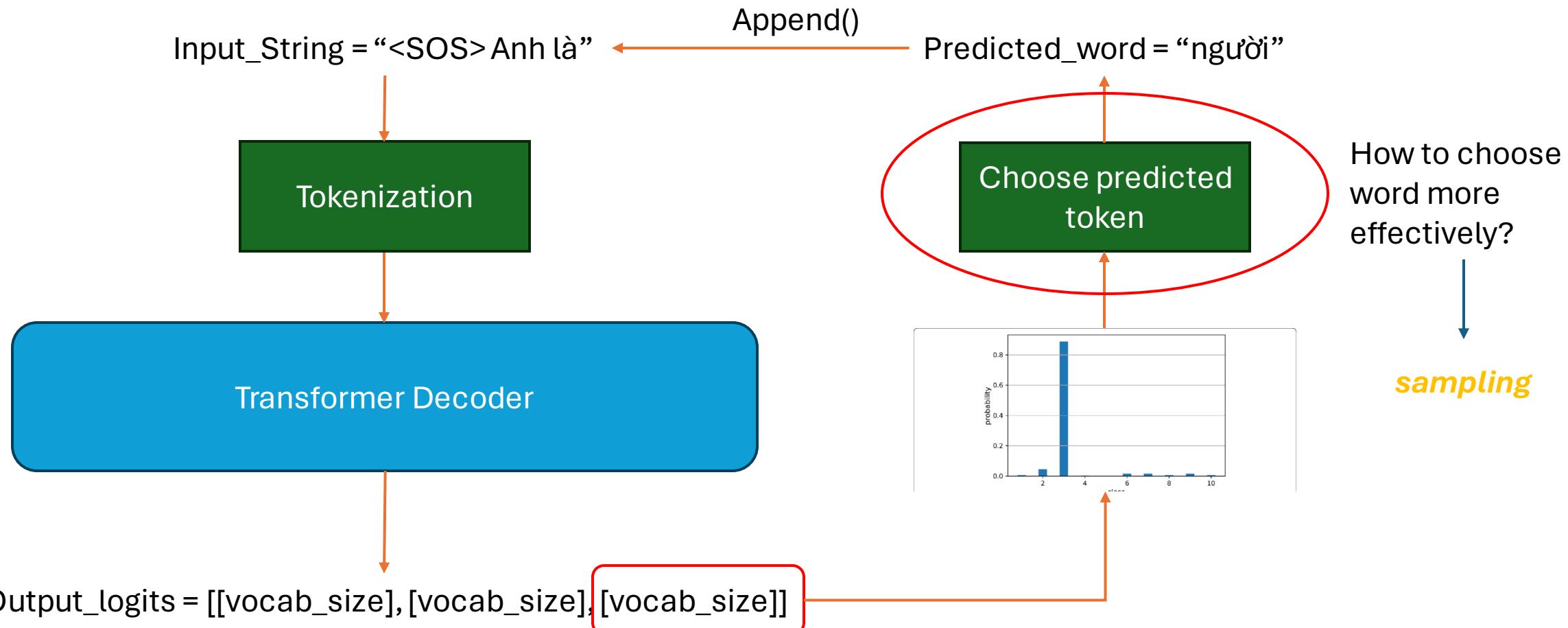
```
1 model.eval()  
2 input_text = '<sos> Anh'  
3 input_tokens = tokenizer(input_text)  
4 input_ids = [vocab[token] for token in input_tokens]  
5 eos_token_id = vocab['<eos>']  
6 generated_ids = input_ids.copy()  
7 MAX_GENERATION_LEN = 50  
8  
9 for _ in range(MAX_GENERATION_LEN):  
10    input_tensor = torch.tensor([generated_ids], dtype=torch.long).to(device)  
11    with torch.no_grad():  
12        outputs = model(input_tensor)  
13  
14    last_token_logits = outputs[0, -1, :]  
15    next_token_id = last_token_logits.argmax().item()  
16    generated_ids.append(next_token_id)  
17  
18    if next_token_id == eos_token_id:  
19        break  
20  
21 # Convert the generated tokens back to text  
22 generated_text = decode(generated_ids)  
23 generated_text = ' '.join(generated_text)  
24 generated_text = generated_text.replace('<sos>', '')  
25 lines = generated_text.split('<eol>')  
26 for line in lines:  
27    print(''.join(line))
```

Anh đi qua đời em những đêm dài
Em dừng vội bâng khuâng
Em dừng vội bâng khuâng

Generate replicated words
(since we always choose highest probability)

Poem Generation: From Scratch

❖ Coding Step 7: Inference (Sampling)

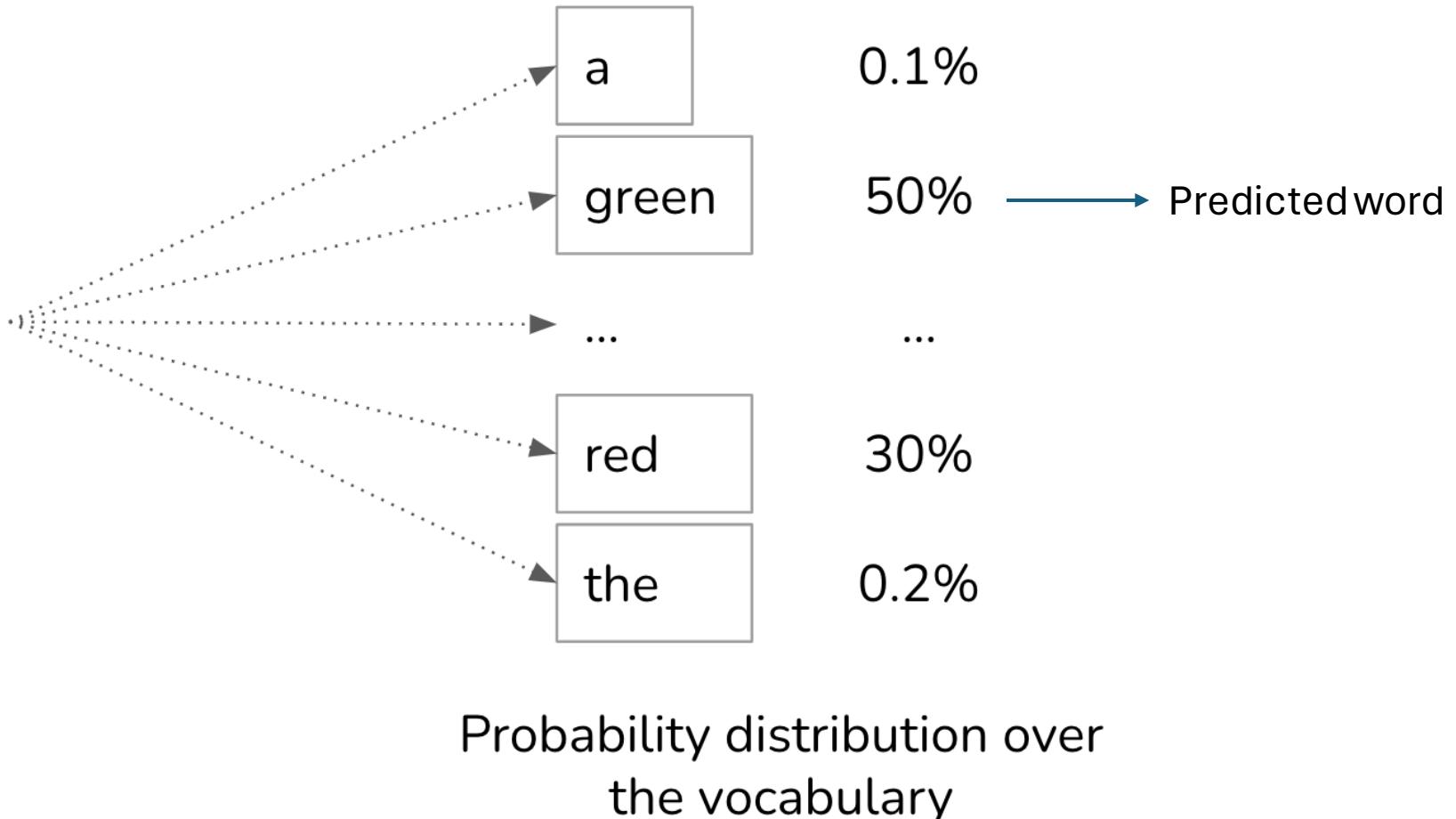


Poem Generation: From Scratch

❖ Coding Step 7: Inference (Sampling)

What's your favorite color?

Greedy sampling
(select highest
probability) (argmax)

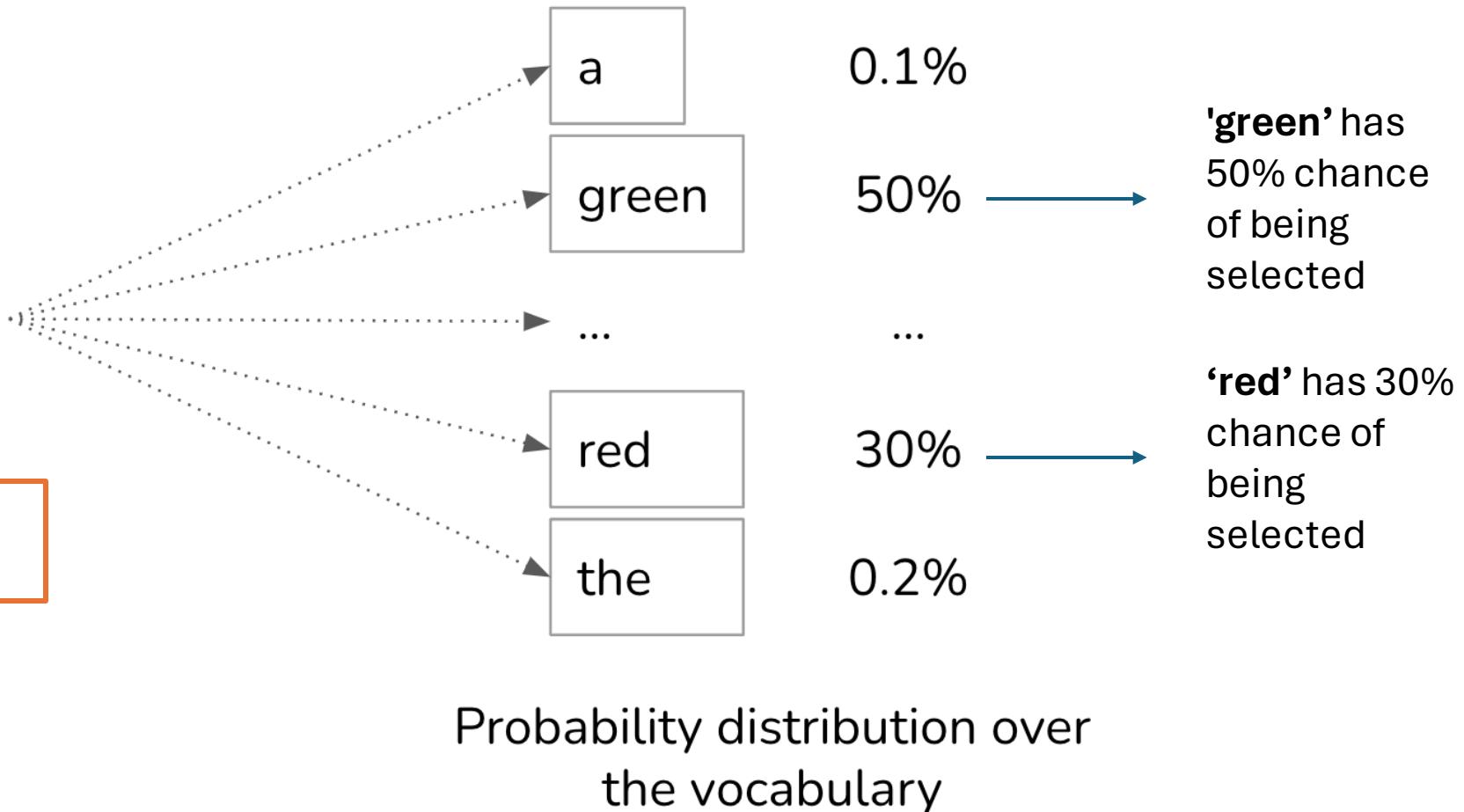


Poem Generation: From Scratch

❖ Coding Step 7: Inference (Sampling)

What's your favorite color?

Random sampling -> Selecting word based on their probability



Poem Generation: From Scratch

❖ Coding Step 7: Inference (Sampling)

```
1 model.eval()
2 input_text = '<sos> Anh'
3 input_tokens = tokenizer(input_text)
4 input_ids = [vocab[token] for token in input_tokens]
5 eos_token_id = vocab['<eos>']
6 generated_ids = input_ids.copy()
7 MAX_GENERATION_LEN = 50
8 for _ in range(MAX_GENERATION_LEN):
9     input_tensor = torch.tensor([generated_ids], dtype=torch.long).to(device)
10    with torch.no_grad():
11        outputs = model(input_tensor)
12
13    last_token_logits = outputs[0, -1, :]
14    next_token_id = sampling(last_token_logits)
15    generated_ids.append(next_token_id)
16
17    if next_token_id == eos_token_id:
18        break
19
20 # Convert the generated tokens back to text
21 generated_text = decode(generated_ids)
22 generated_text = ' '.join(generated_text)
23 generated_text = generated_text.replace('<sos>', '')
24 lines = generated_text.split('<eol>')
25 for line in lines:
26     print(''.join(line))
```

```
1 def sampling(logits):
2     probabilities = F.softmax(logits, dim=-1)
3
4     sampled_index = torch.multinomial(probabilities, 1).item()
5
6     return sampled_index
```

Anh đi qua đồi em được nồng nàn
Chỉ cần có anh bù nhìn
buồn vậy đó
Em dừng vội bâng khuâng
Xông thẳng vào nhau
Săn đuổi mây tháng ròng
Em dừng vội bâng khuâng
Trong chiều thu dịu dàng
Thực ra

Poem Generation: From Scratch

❖ Coding Step 7: Inference (Sampling)

Anh đi qua đời em được nồng nàn
Chỉ cần có anh bù nhìn
buồn vây đó
Em dừng vội bâng khuâng
Xông thẳng vào nhau
Săn đuổi mây tháng ròng
Em dừng vội bâng khuâng
Trong chiều thu dịu dàng
Thực ra

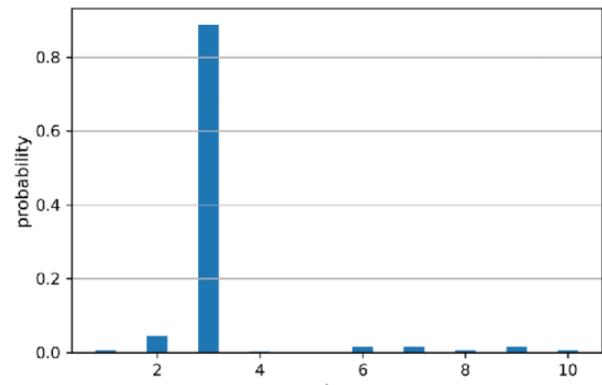


Anh đi qua đời em những đêm dài
Để sớm mai giăng đầy
Nâng niu bằng tâm hồn
Trong chiều thu dịu dàng
dù ngực tù cũng ở điều bình thường
Chỉ cần có anh nữa
sáng soi dòng pháp kệ
Trong chiều thu

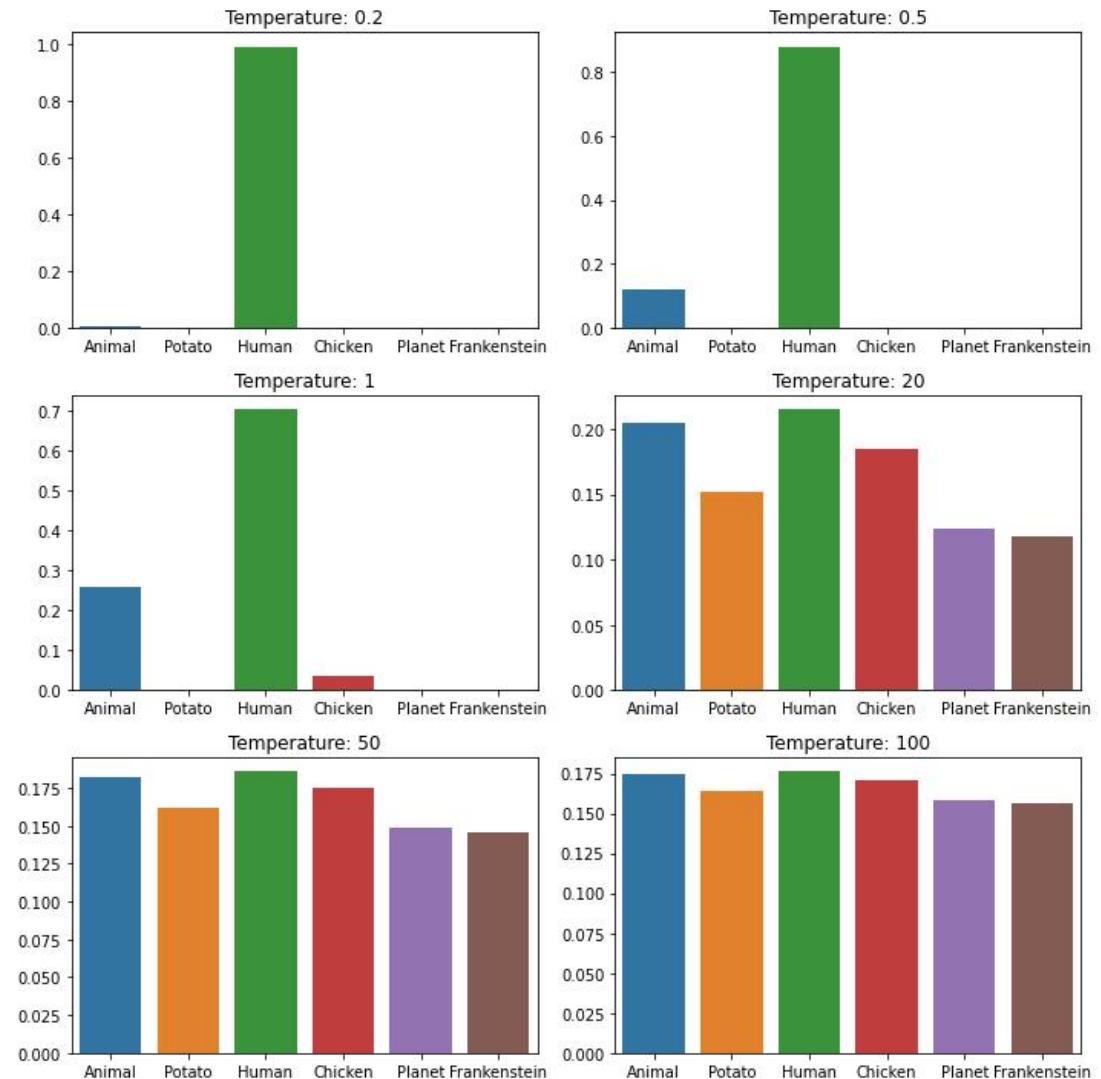
How to make the poem more diverse? => Introduce more randomness?

Poem Generation: From Scratch

❖ Coding Step 7: Inference (Sampling)

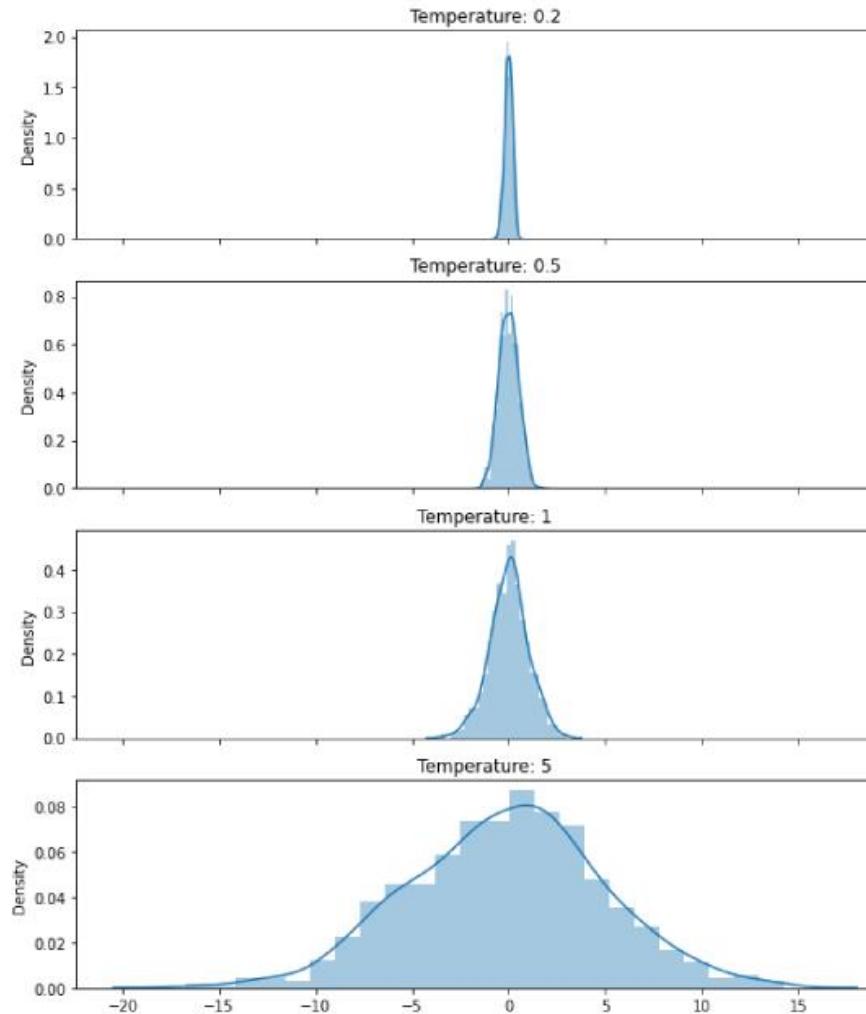


Modify the original probability distribution by “temperature”



Poem Generation: From Scratch

❖ Coding Step 7: Inference (Sampling)



Selecting words based on probability distribution with temperature.

=> **Temperature Sampling**

Poem Generation: From Scratch

❖ Coding Step 7: Inference (Sampling)

```
1 model.eval()
2 temperature = 1.2
3 input_text = '<sos> Anh'
4 input_tokens = tokenizer(input_text)
5 input_ids = [vocab[token] for token in input_tokens]
6 eos_token_id = vocab['<eos>']
7 generated_ids = input_ids.copy()
8 MAX_GENERATION_LEN = 50
9 for _ in range(MAX_GENERATION_LEN):
10     input_tensor = torch.tensor(generated_ids, dtype=torch.long).to(device)
11     with torch.no_grad():
12         outputs = model(input_tensor)
13
14     last_token_logits = outputs[0, -1, :]
15     next_token_id = sample_with_temperature(last_token_logits, temperature=temperature)
16     generated_ids.append(next_token_id)
17
18     if next_token_id == eos_token_id:
19         break
20
21 # Convert the generated tokens back to text
22 generated_text = decode(generated_ids)
23 generated_text = ' '.join(generated_text)
24 generated_text = generated_text.replace('<sos>', '')
25 lines = generated_text.split('<eol>')
26 for line in lines:
27     print(''.join(line))
```

```
1 def sample_with_temperature(logits, temperature=1.0):
2     if temperature != 1.0:
3         logits = logits / temperature
4
5     probabilities = F.softmax(logits, dim=-1)
6
7     sampled_index = torch.multinomial(probabilities, 1).item()
8
9     return sampled_index
```

Anh đi qua đời em những đêm dài
Để sớm mai giăng đầy
Nâng niu bằng tâm hồn
Trong chiều thu dịu dàng
dù nguc tù cũng ở điều bình thường
Chỉ cần có anh nữa
sáng soi dòng pháp kệ
Trong chiều thu

Poem Generation: From Scratch

❖ Coding Step 7: Inference (Sampling)

Anh đi qua đời em những đêm dài
Để sớm mai giăng đầy
Nâng niu bằng tâm hôn
Trong chiều thu dịu dàng
dù ngực tù cũng ở điều bình thường
Chỉ cần có anh nữa
sáng soi dòng pháp kê
Trong chiều thu

Temperature = 1.2

Anh đi qua đời em những đêm dài
Quanh xác thuyền sóng giật
Ơi con chim ột rột
Em đừng vội bâng khuâng
Em đừng vội bâng khuâng
Ơi con chim ột rột
Em đừng vội bâng khuâng
Em đừng vội bâng khuâng

Temperature = 0.5

Poem Generation: GPT2

Poem Generation: GPT2

❖ Introduction

GPT2 (Generative Pre-trained Transformer 2 (GPT-2)) is a large language model developed by OpenAI and the second in their foundational series of GPT models. GPT-2 was pre-trained on BookCorpus, a dataset of over 7,000 self-published fiction books from various genres and trained on a dataset of 8 million web pages. This was proposed in the paper: [Language Models are Unsupervised Multitask Learners](#).

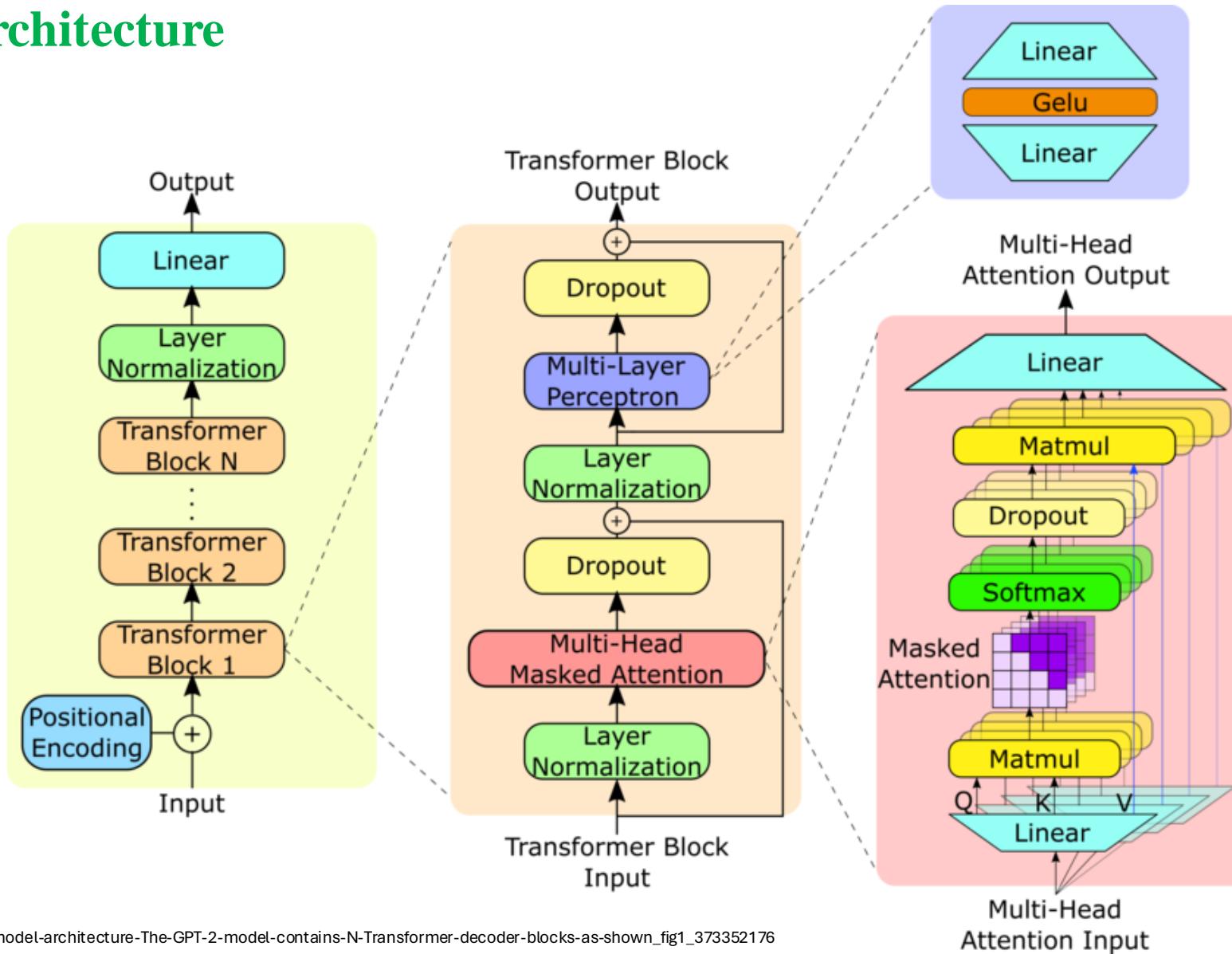
Language Models are Unsupervised Multitask Learners

Alec Radford *¹ Jeffrey Wu *¹ Rewon Child¹ David Luan¹ Dario Amodei **¹ Ilya Sutskever **¹

Model to be used: **GPT2**

Poem Generation: GPT2

❖ GPT2 Architecture



Poem Generation: GPT2

❖ GPT2 on HuggingFace

OpenAI GPT2

All model pages **gpt2** 🤗 Hugging Face Spaces

Overview

OpenAI GPT-2 model was proposed in [Language Models are Unsupervised Multitask Learners](#) by Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei and Ilya Sutskever from [OpenAI](#). It's a causal (unidirectional) transformer pretrained using language modeling on a very large corpus of ~40 GB of text data.

The abstract from the paper is the following:

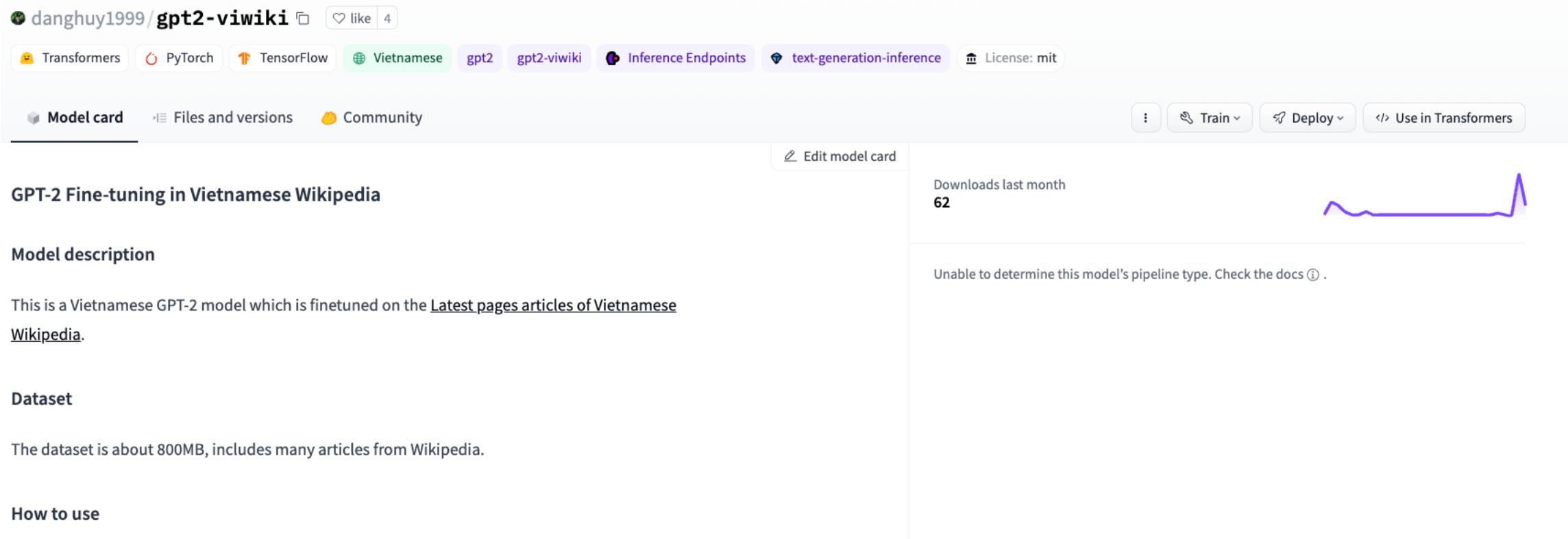
GPT-2 is a large transformer-based language model with 1.5 billion parameters, trained on a dataset[1] of 8 million web pages. GPT-2 is trained with a simple objective: predict the next word, given all of the previous words within some text. The diversity of the dataset causes this simple goal to contain naturally occurring demonstrations of many tasks across diverse domains. GPT-2 is a direct scale-up of GPT, with more than 10X the parameters and trained on more than 10X the amount of data.

[Write With Transformer](#) is a webapp created and hosted by Hugging Face showcasing the generative capabilities of several models. GPT-2 is one of them and is available in five different sizes: small, medium, large, xl and a distilled version of the small checkpoint: *distilgpt-2*.

This model was contributed by [thomwolf](#). The original code can be found [here](#).

Poem Generation: GPT2

❖ Pre-trained Vietnamese GPT2 on HuggingFace



The screenshot shows the HuggingFace Model Card for the 'gpt2-viwiki' model. At the top, it displays the repository name 'danghuy1999/gpt2-viwiki' with 4 likes, and various technology tags: Transformers, PyTorch, TensorFlow, Vietnamese, gpt2, gpt2-viwiki, Inference Endpoints, and text-generation-inference. The license is listed as MIT. Below the header, there are tabs for 'Model card' (which is active), 'Files and versions', and 'Community'. On the right, there are buttons for 'Edit model card', 'Train', 'Deploy', and 'Use in Transformers'. A chart titled 'Downloads last month' shows 62 downloads with a small peak at the end. A note below the chart states: 'Unable to determine this model's pipeline type. Check the docs ⓘ'.

GPT-2 Fine-tuning in Vietnamese Wikipedia

Model description

This is a Vietnamese GPT-2 model which is finetuned on the [Latest pages articles of Vietnamese Wikipedia](#).

Dataset

The dataset is about 800MB, includes many articles from Wikipedia.

How to use

Poem Generation: GPT2

❖ Coding Step 1: Install and import libraries

```
1 !pip install -qq datasets==2.16.1 evaluate==0.4.1 transformers[sentencepiece]==4.35.2
2 !pip install -qq accelerate==0.26.1
3 !apt install git-lfs
```

```
----- 507.1/507.1 kB 10.1 MB/s eta 0:00:00
----- 84.1/84.1 kB 14.8 MB/s eta 0:00:00
----- 7.9/7.9 MB 32.2 MB/s eta 0:00:00
----- 115.3/115.3 kB 18.3 MB/s eta 0:00:00
----- 134.8/134.8 kB 14.0 MB/s eta 0:00:00
----- 134.8/134.8 kB 20.9 MB/s eta 0:00:00
----- 270.9/270.9 kB 5.0 MB/s eta 0:00:00
```

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git-lfs is already the newest version (3.0.2-1ubuntu0.2).
0 upgraded, 0 newly installed, 0 to remove and 35 not upgraded.
```

```
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
```

Poem Generation: GPT2

❖ Coding Step 2: Download and read dataset

```
1 # https://drive.google.com/file/d/1KfrBAycsgQBt1mtEbzMh5pSYL8YIk0Tc/view?usp=sharing
2 !gdown --id 1KfrBAycsgQBt1mtEbzMh5pSYL8YIk0Tc
3 !unzip poem dataset_final.zip
```

```
1 DATASET_PATH = 'datasets/poem_final.csv'  
2 df = pd.read_csv(DATASET_PATH)  
3 df
```

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...	https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1n-Ch%C3%A1m-ngh%C3%B4n-m%C3%9B%C3%A9i-th%C3%9B
1	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...	https://www.thivien.net/L%C3%A1m-Huy-Nhu%E1%BA%ADn-Chieu-co-that-tho
2	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	https://www.thivien.net/Nguy%E1%BB%85n-Nh%E1%BA%A1t-%C4%83nh-My-Mat-bi%C3%A9c
3	3 “Đến, nhiều nơi để đến”	Đến, nhiều nơi để đến\nVề, trở lại với mình\nC...	Nguồn: Châm ngôn mới (thơ), Thái Bá Tân, NXB L...	https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1n-Ch%C3%A1m-ngh%C3%B4n-m%C3%9B%C3%A9i-th%C3%9B
4	4 “Đừng bao giờ dại dột”	Đừng bao giờ dại dột\nĐem chuyện riêng của mìn...	Nguồn: Châm ngôn mới (thơ), Thái Bá Tân, NXB L...	https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%A1n-Ch%C3%A1m-ngh%C3%B4n-m%C3%9B%C3%A9i-th%C3%9B
...

Poem Generation: GPT2

❖ Coding Step 3: Prepare dataset

Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dạy tin yêu.

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lặng lặng,
Trong mơ, ta cùng nhảy.



Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dạy tin yêu.

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lặng lặng,
Trong mơ, ta cùng nhảy.

Our generating objective: Generate 4 lines

Poem Generation: GPT2

❖ Coding Step 3: Prepare dataset

```
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))
```

```
1 print(df['content'][0])
```

```
[['Cái làm ta hạnh phúc', 'Thực ra cũng chẳng nhiều', 'Chỉ cần có ai đó', 'Để ta thảm thương yêu'], ['Rồi thêm chút công việc',
```

Unnamed:	0	title	content
0	0	“Cái làm ta hạnh phúc”	[[Cái làm ta hạnh phúc, Thực ra cũng chẳng nhi...
1	1	“Chiều vừa xốp trên tay”	[[Chiều vừa xốp trên tay, Chợt nghe thoáng ong...
2	2	“Dưới giàn hoa thiên lý”	[[Dưới giàn hoa thiên lý, Một mình anh đang ng...
3	3	“Đến, nhiều nơi để đến”	[[Đến, nhiều nơi để đến, Về, trở lại với mình,...
4	4	“Đừng bao giờ dại dột”	[[Đừng bao giờ dại dột, Đem chuyện riêng của m...
...

Poem Generation: GPT2

❖ Coding Step 3: Prepare dataset

Hạnh phúc trong điều giản,
Ai kia, lòng khẽ hát.
Công việc hàng ngày vui,
Ước mơ dạy tin yêu.

New sample

Đêm thì thầm, trăng dịu,
Bên nhau bước, tình trào.
Sao trên cao, lặng lặng,
Trong mơ, ta cùng nhảy.

New sample

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				
	Unnamed: 0	title	content	source
0	0	"Cái làm ta hạnh phúc"	[Cái làm ta hạnh phúc, Thực ra cũng chẳng nhiề...	Nguồn: Châm ngôn mới (thơ), Thái Bá Tân, NXB L...
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...
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...
3	1	"Chiều vừa xốp trên tay"	[Ớt đỏ sao cù đỏ, Táo chín cho thật vàng, Em đ...	Nguồn: Lâm Huy Nhuận, Chiều có thật (thơ), NXB...
4	2	"Dưới giàn hoa thiên lý..."	[Dưới giàn hoa thiên lý, Một mình anh đang ngồ...	Nguồn: Nguyễn Nhật Ánh, Mắt biếc, NXB Trẻ, 2004
...

Poem Generation: GPT2

❖ Coding Step 3: Prepare dataset

```
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...	https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%AD%C3%A1n
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...	https://www.thivien.net/Th%C3%A1i-B%C3%A1-T%C3%AD%C3%A1n
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...	https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA...
3	1	“Chiều vừa xốp trên tay”	Ót đở sao cứ đở\nTáo chín cho thật vàng\nEm đẹ...	Nguồn: Lâm Huy Nhuận, Chiều có thật (thơ), NXB...	https://www.thivien.net/L%C3%A2m-Huy-Nhu%E1%BA...
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	https://www.thivien.net/Nguy%E1%BB%85n-Nh%E1%BA...
...

Poem Generation: GPT2

❖ Coding Step 3: Prepare dataset

```
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
    })
})
```

Poem Generation: GPT2

❖ Coding Step 4: Preprocess dataset

```
1 MODEL_NAME = 'danghuy1999/gpt2-viwiki'  
2  
3 tokenizer = GPT2Tokenizer.from_pretrained(MODEL_NAME)  
  
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:88: UserWarning:  
The secret `HF_TOKEN` does not exist in your Colab secrets.  
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret  
You will be able to reuse this secret in all of your notebooks.  
Please note that authentication is recommended but still optional to access public models or datasets.  
    warnings.warn(  
vocab.json: 100%  773k/773k [00:00<00:00, 3.93MB/s]  
merges.txt: 100%  431k/431k [00:00<00:00, 3.33MB/s]  
config.json: 100%  916/916 [00:00<00:00, 69.1kB/s]
```

Pre-trained Vietnamese GPT2: [Link](#)

Poem Generation: GPT2

❖ Coding Step 4: Preprocess dataset

```
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 )
```

```
1 tokenizer('xin chào các bạn', max_length=10, padding='max_length', truncation=True)
{'input_ids': [6504, 3939, 331, 1486, 0, 0, 0, 0, 0, 0], 'attention_mask': [1, 1, 1, 1, 0, 0, 0, 0, 0, 0]}

1 tokenized_poem_dataset['train'][0].keys()
dict_keys(['input_ids', 'attention_mask'])

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

Poem Generation: GPT2

❖ Coding Step 5: Training

```
1 model = GPT2LMHeadModel.from_pretrained(MODEL_NAME)

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     push_to_hub=True
9 )
```

```
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()

[1000/1000 05:39, Epoch 20/20]
```

Step	Training Loss
500	5.629400
1000	4.272700

Poem Generation: GPT2

❖ Coding Step 6: Inference

```
1 from transformers import pipeline
2
3 prompt = 'Con sông quê tôi đẹp\n'
4 generator = pipeline('text-generation', model='thangduong0509/gpt2_viet_poem_generation')
5 results = generator(
6     prompt,
7     max_new_tokens=50,
8     do_sample=True,
9     top_k=50,
10    top_p=0.95,
11    temperature=0.8,
12    repetition_penalty=1.2
13 )
14 results = results[0]['generated_text']
15 print()
16 for line in results.split('\n'):
17     print(line)
```

Con sông quê tôi đẹp
Chảy xa đâu nắng!
Người bạn tình yêu...
Một cách nào nào thì?
Có thể đến rất nhiều!
Nên đất nước nhỏ... còn nhỏ?
Làm thật là thê!
Đời mình

Poem Generation: GPT2

❖ Coding Step 6: Inference

```
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)
```

Học học nữa học mãi
Hàng trăm ngàn đêm dài
Nhưng, không ai quên
Quanh ta thấy chuyện? đở?
Tôi thấy mình là học giả!
Đôi hay bạc đau may chờ.
Còn để ra mắt nào nữa?
Vì, ta thấy bạn

Question

