DATA_301_Assignment_04_ISHAAN_SATHAYE_SRESHTA_TALLURI

November 2, 2023

1 Song Lyrics Generator

In this assignment, you will scrape a website to get lyrics of songs by your favorite artist. Then, you will train a model called a Markov chain on these lyrics so that you can generate a song in the style of your favorite artist.

2 Question 1. Scraping Song Lyrics

Find a web site that has lyrics for several songs by your favorite artist. Scrape the lyrics into a Python list called `lyrics`, where each element of the list represents the lyrics of one song.

Tips: - Find a web page that has links to all of the songs, like this one. Then, you can scrape this page, extract the hyperlinks, and issue new HTTP requests to each hyperlink to get each song. - If you can't find the artist or songs you want on https://www.songlyrics.com/ you can try some of the music related APIs here. If you find a useful site, please share it with everyone on Discord. - Use `time.sleep()` to stagger your HTTP requests so that you do not get banned by the website for making too many requests.

[5]:
```
import requests
import time

from bs4 import BeautifulSoup

response = requests.get("https://www.songlyrics.com/jess-glynne-lyrics/")

soup = BeautifulSoup(response.content, 'html.parser')
```

[6]:
```
song_table = soup.find_all('table', attrs={'class': 'tracklist'})[0]
len(song_table)
```

[6]: 3

[7]:
```
song_links = []

for song in song_table.find_all("a"):

```

```
    # Get the link for the song
    link = song.get('href')

    # Append this data.
    song_links.append(link)

len(song_links)
```

[7]: 71

[8]:
```
unclean_lyrics = []

for link in song_links:
    response = requests.get(link)
    soup = BeautifulSoup(response.content, 'html.parser')
    unclean_lyrics.append(soup.find_all('p', attrs={'id': 'songLyricsDiv'})[0].
 ↪text)
    time.sleep(0.5)
```

[9]:
```
import re

lyrics = []
for lyric in unclean_lyrics:
    if "We do not have the lyrics for" not in lyric:
        lyrics.append(lyric)
```

[10]:
```
# Print out the lyrics to the first song.
print(lyrics[0])
```

```
Standing in a crowded room and I can't see your face
Put your arms around me, tell me everything's OK
In my mind, I'm running round a cold and empty space
Just put your arms around me, tell me everything's OK
Break my bones but you won't see me fall, oh
The rising tide will rise against them all, oh

Darling, hold my hand
Oh, won't you hold my hand?
Cause I don't wanna walk on my own anymore
Won't you understand? Cause I don't wanna walk alone
I'm ready for this, there's no denying
I'm ready for this, you stop me falling
I'm ready for this, I need you all in
I'm ready for this, so darling, hold my hand
Soul is like a melting pot when you're not next to me
Tell me that you've got me and you're never gonna leave
```

```
Tryna find a moment where I can find release
Please tell me that you've got me and you're never gonna leave
Break my bones but you won't see me fall, oh
The rising tide will rise against them all, oh

Darling, hold my hand
Oh, won't you hold my hand?
Cause I don't wanna walk on my own anymore
Won't you understand? Cause I don't wanna walk alone
I'm ready for this, there's no denying
I'm ready for this, you stop me falling
I'm ready for this, I need you all in
I'm ready for this, so darling, hold my hand

Don't wanna know
That feeling when I'm all alone
So please don't make me wait, cause I don't wanna break
And I don't wanna fall
When you're next to me
Can tell I'm not afraid to be
That you don't make me wait, and never let me break
You never let me fall
```

`pickle` is a Python library that serializes Python objects to disk so that you can load them in later.

[11]:
```
import pickle
pickle.dump(lyrics, open("lyrics.pkl", "wb"))
```

3 Question 2. Unigram Markov Chain Model

You will build a Markov chain for the artist whose lyrics you scraped in Question 1. Your model will process the lyrics and store the word transitions for that artist. The transitions will be stored in a dict called `chain`, which maps each word to a list of "next" words.

For example, if your song was "The Joker" by the Steve Miller Band, `chain` might look as follows:

```
chain = {
    "some": ["people", "call", "people"],
    "call": ["me", "me", "me"],
    "the": ["space", "gangster", "pompitous", ...],
    "me": ["the", "the", "Maurice"],
    ...
}
```

Besides words, you should include a few additional states in your Markov chain. You should have `"<START>"` and `"<END>"` states so that we can keep track of how songs are likely to begin and end. You should also include a state called `"<N>"` to denote line breaks so that you can keep track of where lines begin and end. It is up to you whether you want to include normalize case and strip

punctuation.

So for example, for "The Joker", you would add the following to your chain:

```
chain = {
    "<START>": ["Some", ...],
    "Some": ["people", ...],
    "people": ["call", ...],
    "call": ["me", ...],
    "me": ["the", ...],
    "the": ["space", ...],
    "space": ["cowboy,", ...],
    "cowboy,": ["yeah", ...],
    "yeah": ["<N>", ...],
    "<N>": ["Some", ..., "Come"],
    ...,
    "Come": ["on", ...],
    "on": ["baby", ...],
    "baby": ["and", ...],
    "and": ["I'll", ...],
    "I'll": ["show", ...],
    "show": ["you", ...],
    "you": ["a", ...],
    "a": ["good", ...],
    "good": ["time", ...],
    "time": ["<END>", ...],
}
```

Your chain will be trained on not just one song, but by all songs by your artist.

[12]:
```
def train_markov_chain(lyrics):
    """
    Args:
      - lyrics: a list of strings, where each string represents
                the lyrics of one song by an artist.

    Returns:
      A dict that maps a single word ("unigram") to a list of
      words that follow that word, representing the Markov
      chain trained on the lyrics.
    """
    chain = {"<START>": [], "<N>": []}
    for lyric in lyrics:
      # YOUR CODE HERE
      lines = lyric.split("\n")
      lines = [line for line in lines if line != ""]
      for l in range(len(lines)):
        words = lines[l].split(" ")
        # remove all punctuation
```

```
        words = [re.sub(r'[^\w\s]','',word) for word in words]
        # remove \r characters
        words = [word.replace("\r", "") for word in words]
        # add start and end tokens
        if l == 0:
          words = ["<START>"] + words + ["<N>"]
        elif l == len(lines)-1:
          words = ["<N>"] + words + ["<END>"]
        else:
          words = ["<N>"] + words + ["<N>"]
        for i in range(len(words)-1):
          if words[i] not in chain:
            chain[words[i]] = []
          chain[words[i]].append(words[i+1])

    return chain
```

[13]:
```
# Load the pickled lyrics object that you created in Question 1.
import pickle
lyrics = pickle.load(open("lyrics.pkl", "rb"))

# Call the function you wrote above.
uni_chain = train_markov_chain(lyrics)

# What words tend to start a song (i.e., what words follow the <START> tag?)
print(uni_chain["<START>"])
```

```
['Standing', 'Standing', 'Theres', 'Standing', 'Theres', 'Téléchargez', 'From',
'Wrapped', 'I', 'Téléchargez', 'Téléchargez', 'Finally', 'Finally', 'Finally',
'Finally', 'Finally', 'Finally', 'When', 'When', 'Standing', 'Standing',
'Standing', 'Wrapped', 'in', 'I', 'I', 'I', 'Wrapped', 'Wrapped', 'Téléchargez',
'Finally', '', 'feat', 'with', 'feat', 'Verse', 'Verse', 'Standing', 'Standing',
'Standing', 'Standing', 'Sometimes', 'Theres', 'Going', 'You', 'Smoking',
'Time', 'Thinking', 'Wrapped', 'I', 'I', 'Birds', 'In', 'Dont', 'Theres',
'with']
```

Now, let's generate new lyrics using the Markov chain you constructed above. To do this, we'll begin at the `"<START>"` state and randomly sample a word from the list of words that follow `"<START>"`. Then, at each step, we'll randomly sample the next word from the list of words that followed each current word. We will continue this process until we sample the `"<END>"` state. This will give us the complete lyrics of a randomly generated song!

You may find the `random.choice()` function helpful for this question.

[14]:
```
import random

def uni_generate_new_lyrics(chain):
    """
```

```
    Args:
      - chain: a dict representing the Markov chain,
               such as one generated by generate_new_lyrics()

    Returns:
      A string representing the randomly generated song.
    """

    # a list for storing the generated words
    words = []
    # generate the first word
    words.append(random.choice(chain["<START>"]))

    # # YOUR CODE HERE
    while words[-1] != "<END>":
        choices = chain.get(words[-1], ["<END>"])
        words.append(random.choice(choices))

    # # join the words together into a string with line breaks
    lyrics = " ".join(words[:-1])
    return "\n".join(lyrics.split("<N>"))
```

[15]:
```
print(uni_generate_new_lyrics(uni_chain))
```

```
Finally Im right here you were wrong if I cant give you take me so darling hold
my life would end and never gonna leave
 Wont you talk to think its what I wouldnt feel it go she call from you I know
 Right here you dont make it
 I dont be there for wishing not wrong
 And I know that no
 PreChorus Jess Glynne x2
 You can see your calling
 Ooh oh
 So dont wanna walk on yourself no one for this black heart turn this you dont
you with the world now Im running all I wouldnt change it from underneath my
hand
 Wont you all
 If I wanna walk alone with you want nobody baby
 Oh will heal
 Its time
 Why me where you give one for your spirit fading in
 Take me home
 Soul is made it aint got my hand
 Ill be there for this darkness over just like Ive landed on yourself no denying
 Anger love and let you hold me walk on the rhythm of mine

 Something that I hit the way you want me
```

Infatuation took a moment where you I cant have you my own
And nobodys stone
So I wont be alone
Everything thats broke my
Oh will make it go go
Hold the wheel
Im ready for this you something I dont know I I spent my rose
No rights no denying
Oh yes about that never knew where Id be there for you take the saddest vanilla that no longer a dream
Im not next to walk alone
If I wanna walk on my tears dry air to you least expect its in a hole of my hand
So Ima give me
Dont you I cant shake me down
Oh will find me Im not letting go leaving pole position
That youll be there for me
So Ima give me
But hearts break and youre quite shy
I to think I spent my eyes
Im not to my bones
Wont let my darkest thoughts

Im content
If I wanna walk alone
If I didnt think I got me where I know
Right here right and empty space
Like Im haunted oh ah ah ah ah
You gave me and over mine
Take it from sayin no denying
Everything thats broke
Right here you wrong
Why me
Now youre never gonna be
If you take the hook but I cant have you oh oh oh
You left me love and hells a thousand miles from me fall oh oh oh oh oh
Dont be you I need you got a broken soul
And now
Lets go back home
Now Im content
Lets go go go
Im ready for this on let my tears fill my touch
Ooh oh
Why dont make it cause I cant have you with you something we turned this I would take me
Break my heart my life was right no it worked for this black heart made it to you blazing you with heaven youre never be so hard on So dont need
I didnt have you talk to you hold

Now Im right no no it go go and I dont be there eventually
So dont make it heal
And it will find mine
Like Im on that
I dont be for this you my darkest thoughts
Something must be
To staying up above made with the floor
But hearts break and you got me
And I never good
Right here you want me falling
Cause I wouldnt change it aint got me now
Ill be there when Im right and youre never knew where you hold my hand
My life
And when your calling
You left me here waiting for you with a promise Never broke
We have you want nobody baby
Now Im ready for this I feel it just like Ive landed on me calling
Im ready for this you
Looovve
Oh oh oh
And now
Ooh oh oh
And then its no wrongs take me that feeling you wont you talk to
Its a blessing
If I aint been one for two ooh
Oh will make me walk away am I dont wanna LEAVE
Now youre not letting go back to see me
If I wouldnt feel it all along
Kind of my touch up above is made with the warmth of my bones but Ill be take me like Ive got me love
When all so darling hold my hand
If I lose control
Cause the warmth of frail I didnt think I dont wanna be lost forever
Its not complaining
Cant let go back to let it go for this darkness over me

You gave me now were wrong for you there
That feeling that you take it for love and wait cause you least expect its creepin up on my bones
Right here waiting for this you still wont see it to simplicity
My love me
Oh oh oh falling
But hearts break
Now Im right here right here
Right here you with none Im right here you can this so hard on yourself no
Learn to learn to stay
Cause you baby

```
You say space
Now Im not an open end and never let it be
Cause I dont sleep tonight
Soul is like a smile on yourself no more you
I lose time
Oh wont be so hard on my eyes be there
I thought you all of gold
When theres no rules
Oh I dont make it from me that love
```

4 Question 3. Bigram Markov Chain Model

Now you'll build a more complex Markov chain that uses the last *two* words (or bigram) to predict the next word. Now your dict `chain` should map a *tuple* of words to a list of words that appear after it.

As before, you should also include tags that indicate the beginning and end of a song, as well as line breaks. That is, a tuple might contain tags like `"<START>"`, `"<END>"`, and `"<N>"`, in addition to regular words. So for example, for "The Joker", you would add the following to your chain:

```
chain = {
    (None, "<START>"): ["Some", ...],
    ("<START>", "Some"): ["people", ...],
    ("Some", "people"): ["call", ...],
    ("people", "call"): ["me", ...],
    ("call", "me"): ["the", ...],
    ("me", "the"): ["space", ...],
    ("the", "space"): ["cowboy,", ...],
    ("space", "cowboy,"): ["yeah", ...],
    ("cowboy,", "yeah"): ["<N>", ...],
    ("yeah", "<N>"): ["Some", ...],
    ("time", "<N>"): ["Come"],
    ...,
    ("<N>", "Come"): ["on", ...],
    ("Come", "on"): ["baby", ...],
    ("on", "baby"): ["and", ...],
    ("baby", "and"): ["I'll", ...],
    ("and", "I'll"): ["show", ...],
    ("I'll", "show"): ["you", ...],
    ("show", "you"): ["a", ...],
    ("you", "a"): ["good", ...],
    ("a", "good"): ["time", ...],
    ("good", "time"): ["<END>", ...],
}
```

[16]:
```python
def train_markov_chain(lyrics):
    """
    Args:
```

```
      - lyrics: a list of strings, where each string represents
                the lyrics of one song by an artist.

    Returns:
      A dict that maps a tuple of 2 words ("bigram") to a list of
      words that follow that bigram, representing the Markov
      chain trained on the lyrics.
    """
    chain = {(None, "<START>"): []}
    for lyric in lyrics:
        words = lyric.split(" ")
        # remove all punctuation
        words = [re.sub(r'[^\w\s]','',word) for word in words]
        # remove \r characters
        words = [word.replace("\r", "") for word in words]
        words = [None, "<START>"] + words + ["<END>"]
        for i in range(len(words)-2):
            bigram = (words[i], words[i+1])
            if bigram not in chain:
                chain[bigram] = []
            chain[bigram].append(words[i+2])
    return chain
```

[17]:
```
# Load the pickled lyrics object that you created in Question 1.
import pickle
lyrics = pickle.load(open("lyrics.pkl", "rb"))

# Call the function you wrote above.
bi_chain = train_markov_chain(lyrics)

# What words tend to start a song (i.e., what words follow the <START> tag?)
print(bi_chain[(None, "<START>")])
```

```
['Standing', 'Standing', 'Theres', 'Standing', 'Theres', 'Téléchargez', 'From',
'Wrapped', 'I', 'Téléchargez', 'Téléchargez', 'Finally', 'Finally', 'Finally',
'Finally', 'Finally', 'Finally', 'When', 'When', 'Standing', 'Standing',
'Standing', 'Wrapped', 'in', 'I', 'I', 'I', 'Wrapped', 'Wrapped', 'Téléchargez',
'Finally', '', 'feat', 'with', 'feat', 'Verse', 'Verse', 'Standing', 'Standing',
'Standing', 'Standing', 'Sometimes', 'Theres', 'Going', 'You', 'Smoking',
'Time', 'Thinking', 'Wrapped', 'I', 'I', 'Birds', 'In', 'Dont', 'Theres',
'with']
```

Now, let's generate new lyrics using the Markov chain you constructed above. To do this, we'll begin at the `(None, "<START>")` state and randomly sample a word from the list of words that follow this bigram. Then, at each step, we'll randomly sample the next word from the list of words that followed the current bigram (i.e., the last two words). We will continue this process until we sample the `"<END>"` state. This will give us the complete lyrics of a randomly generated song!

[18]:
```
import random

def bi_generate_new_lyrics(chain):
    """
    Args:
      - chain: a dict representing the Markov chain,
               such as one generated by generate_new_lyrics()

    Returns:
      A string representing the randomly generated song.
    """

    # a list for storing the generated words
    words = []
    # generate the first word
    bigram = (None, "<START>")
    next = random.choice(chain[bigram])
    words.append(next)
    bigram = (bigram[1], next)

    # YOUR CODE HERE
    while bigram[1] != "<END>":
        choices = chain.get(bigram, ["<END>"])
        next = random.choice(choices)
        words.append(next)
        bigram = (bigram[1], next)

    # join the words together into a string with line breaks
    lyrics = " ".join(words[:-1])
    return "\n".join(lyrics.split("<N>"))
```

[26]:
```
print(bi_generate_new_lyrics(bi_chain))
```

```
Birds fly we turned finally free
Patience lost I began to lose me
My advice would be take a step

PreChorus
I wasnt scared I fought this on my own anymore wont you hold me now
Oh will you take me home
Oh will you take me home
You say space will make it hard to get for a question
Not enough words to make a sentence
Its not easy to find our inner peace
Make it everlasting so nothings incomplete
Its easy being with you when your hair get thinner
I aint gotta work it out I know I love it when I see a break in the dark
```

Oh oh Ill be there for you
Ill be there

Ill be there for you
Ill be there Ill be there
When you need a little love to share
Yeah Im gonna Im gonna come through
Youll never be alone Ill be there for you
Ill be there cant you hear me calling
Oh I swear I got a call from you
I wont confess

And I I wont be lost forever
And soon I wouldnt want you to know
And I I know I know I know
That I aint got far to go

Chorus
I know I know
I know I Know
I Know I know I know I Know
I Know I know I know oh oh oh
Aah ah ah ah
Ooh oh oh ah ah ah
Oh oh oh

Infatuation took a whole of my love
Looovve

Control is such an openended word for me
Something that I used to no rules
And now Im here with you

Right here you got me where you want me
If you want me
Now Im right here right here

Ooh oh oh oh
Aah ah ah ah
Oh oh oh

Finally Im content
Oh yes about that thing
Right here is where Id stay
But Im not afraid for I will do

If I cant have you

Paste your randomly generated song lyrics (either unigram or bigram) into the Discord channel and we can try to guess the artist!

5 Question 4. Analysis

Compare the quality of the lyrics generated by the unigram model (in Question 2) and the bigram model (in Question 3). Which model seems to generate more reasonable lyrics? Can you explain why? What do you see as the advantages and disadvantages of each model?

YOUR ANSWER HERE.

From comparing the quality of lyrics of the unigram model in question 2 and the bigram model in question 3, the bigram seems to have better cohesion and flow. This is because the bigram model takes into account the previous word and the current word to predict the next word. Also the bigram takes into account context, since it takes in 2 words. However, the unigram only takes the next word into account, leading to less coherent lyrics.

The advantage of the unigram model is that it is easier to implement and takes less time to run. The disadvantage of the unigram model is that it does not take into account the previous word to predict the next word. Bigram would be better to use for generating lyrics since it takes into account the word choices of the artist and the context of the lyrics, leading to similar song styles.

5.1 Submission Instructions

- After you have completed the notebook, select **Runtime > Run all**
- After the notebook finishes rerunning check to make sure that you have no errors and everything runs properly. Fix any problems and redo this step until it works.
- Rename this notebook by clicking on "DATA 301 Assignment 04 - YOUR NAMES HERE" at the very top of this page. Replace "YOUR NAMES HERE" with the first and last names of you and your partner (if you worked with one).
- Expand all cells with View > Expand Sections
- Save a PDF version: File > Print > Save as PDF
  - Under "More Settings" make sure "Background graphics" is checked
  - Printing Colab to PDF doesn't always work so well and some of your output might get cutoff. That's ok.
  - It's not necessary, but if you want a more nicely formatted PDF you can uncomment and run the code in the following cell. (Here's a video with other options.)
- Download the notebook: File > Download .ipynb
- Submit the notebook and PDF in Canvas. If you worked in a pair, only one person should submit in Canvas.

[20]:
```
# !wget -nc https://raw.githubusercontent.com/brpy/colab-pdf/master/colab_pdf.py
# from colab_pdf import colab_pdf
# colab_pdf('DATA 301 Lab4B - YOUR NAMES HERE.ipynb')
```