Poetry Generation in Urdu

Muhammad Usman P19-0096

February 24, 2023

1 Introduction

Poetry Generation in Urdu using n-gram language modeling to generate some poetry using the spaCy library for text processing.

2 Load all poetry file in single string

```
import os

# Path to directory containing text files
directory = "/home/usman/Documents/Study/NLP/Assign-2"

# Generate a list of file paths for all the text files
file_paths = [os.path.join(directory, file) for file in os.listdir(directory)
if file.endswith(".txt")]

corpus = {}
for file_path in file_paths:
with open(file_path, "r") as f:
text = f.read()
corpus[file_path] = text

# Combine all the text data into a single string
full_text = " ".join(corpus.values())
```

This code reads text data from multiple text files in a directory and creates a corpus dictionary where each key is a file path and the corresponding value is the text content of the file. It then combines all the text data into a single string full-text

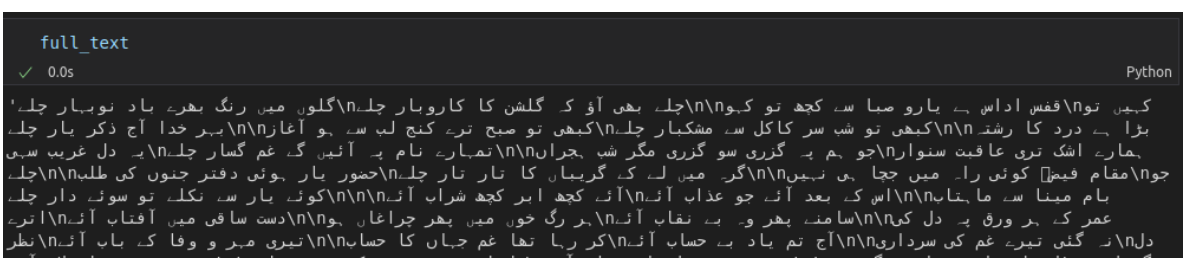

Figure 1: This is Output of the loaded corpus

2.1 Poetry Generation Code

```
import spacy
import random

# Load the language model
nlp = spacy.blank('ur')

# The poem will have three stanzas, each containing four verses
num_stanzas = 3
num_verses = 4

# Set the value of n for n-grams
n = 2

text=full_text
# Preprocess the text
text = text.lower()
text = text.replace('\n', ' ')

# Tokenize the text
doc = nlp(text)
words = [token.text for token in doc]

# Create n-grams
ngrams = {}
for i in range(len(words)-n):
gram = ' '.join(words[i:i+n])
if gram not in ngrams.keys():
ngrams[gram] = []
ngrams[gram].append(words[i+n])

# Generate the poem
poem = ''
for i in range(num_stanzas):
for j in range(num_verses):
current_gram = random.choice(list(ngrams.keys()))
verse = current_gram.capitalize()
for k in range(7-n):
if current_gram not in ngrams.keys():
break
possible_words = ngrams[current_gram]
next_word = possible_words[random.randrange(len(possible_words))]
verse += ' ' + next_word
words = current_gram.split(' ')
words.append(next_word)
current_gram = ' '.join(words[1:])
poem += verse + '\n'
poem += '\n'

print(poem)
```

This part of code generates a poem using n-gram language modeling. The poem has three stanzas, each containing four verses. The value of n for the n-gram model is set to 2.

Loads a blank Urdu language model from the spaCy library, preprocesses the input text, tokenizes

it, and creates n-grams from the tokens. It then generates the poem by randomly choosing an n-gram and selecting the next word based on the probability distribution of words that follow that n-gram in the input text. This process is repeated to generate each verse of the poem.

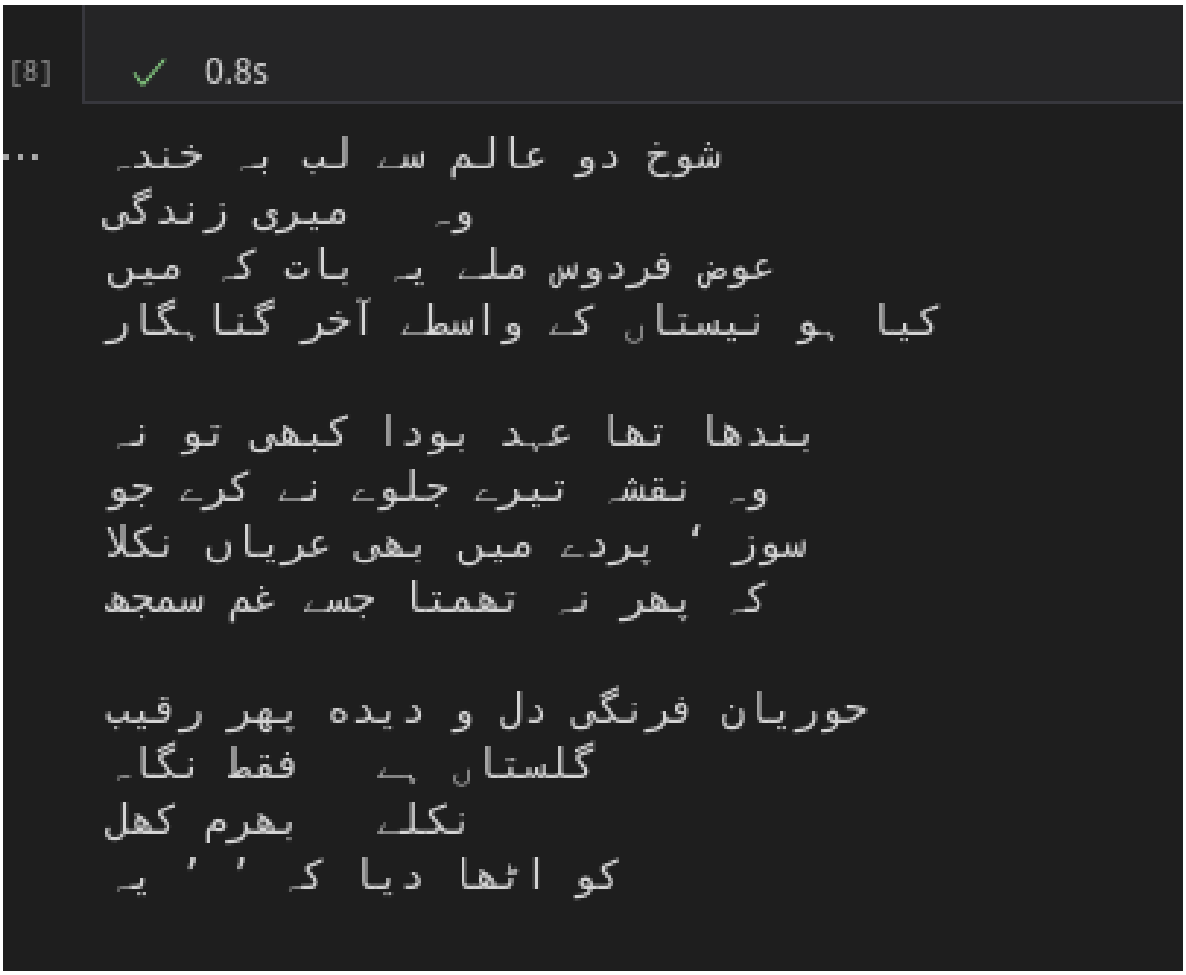

Figure 2: This is Output of the Predicted poem