TITLE OF PROJECT

NAMED ENTITY RECOGNITION

END TERM REPORT

by

NAME OF THE CANDIDATE(S)

Uppala manoj
Charapalli tharun
Yarakaraju ravi varma
Asireddy Vivekananda reddy

SECTION: K18SJ
ROLL NUMBER(S): 47,46,40,39

Department of Intelligent Systems

School of Computer Science Engineering

Lovely Professional University, Punjab

April-2020

APPENDIX 2

Student Declaration

This is to declare that this report has been written by me/us. No part of the report is copied from other sources. All information included from other sources have been duly acknowledged. I/We aver that if any part of the report is found to
be copied, I/we are shall take full responsibility for it.

Uppala manoj
Roll number: 47 (11812766)

Charapalli tharun
Roll number: 46 (11803163)

Yarakaraju ravi varma
Roll number: 40 (11814445)

Asireddy Vivekananda reddy
Roll number: 39 (11802264)

Lovely Professional University, Punjab

APPENDIX 3

TABLE OF CONTENTS

BONAFIDE CERTIFICATE

Certified that this project report "**GENERATION OF MUSIC USING LSTM NEURAL NETWORK"** is the bonafide work of "**Uppala Manoj, Charapalli Tharun, Yarakaraju ravi varma, Asireddy vivekananda reddy**" who carried out the project work under my supervision.

Mr. Dipen Saini
23681
Department of Intelligent Systems
School of Computer Science Engineering
Lovely Professional University, Punjab

1.) Introduction and Objective

People, places, and things—nouns—play a crucial role in language, conveying the sentence's subject and often its object. Due to their importance, it's often useful when processing text to try to identify nouns and use them in applications. This is mostly known as either entity identification or named-entity recognition (NER) which is often handled by a parser or chunker. Though using a parser is nice for understanding a sentence, text applications often will find it more useful to focus on a subset of nouns that identify specific instances of an object such as proper nouns, also known mostly as entities. In many situations, it's also useful to go beyond names of people and places and include temporal and numeric concepts like August 2019 as date or $50.35 as price.

In this project we have used nltk module to extract entities from a sentence and categorize the entities as person, place or thing. Generating a Parse tree, calculating number of each parts of speech are also the features of this project.

2.) Knowledge

Following are few terms that are used in named entity recognition one should know before moving further.

2.1) Tokenization

Tokenization is the process of chopping the given text into smaller and most understandable units called tokens. Tokens can either be words, sentences, etc.

Tokenization is the process of demarcating and possibly classifying sections of a string of input characters. The resulting tokens are then passed on to some other form of processing. The process can be considered a sub-task of parsing input.

For example, in the text string:

```
The quick brown fox jumps over the lazy dog
```

the string isn't implicitly segmented on spaces, as a natural language speaker would do. The raw input, the 43 characters, must be explicitly split into the 9 tokens with a given space delimiter.

```
In [2]: import nltk
        from nltk import word_tokenize
        print(word_tokenize('the quick brown fox jumps over the lazy dog'))

['the', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog']
```

2.2) parts of speech

In traditional grammar, a **part of speech** is a category of words (or, more generally, of lexical items) that have similar grammatical properties. Words that are assigned to the same part of speech generally display similar syntactic behavior—they play similar roles within the grammatical structure of sentences—and sometimes similar morphology in that they undergo inflection for similar properties.

2.3) NLTK

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora languages such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Project Description

3.1.) Processing the text

3.1.1) Tokenization of text

The text that is entered in the textbox in passed to the word_tokenization function which generates tokens. The word_tokenize function returns a list of strings. The sent_tokenize function can be used to extract sentences from a paragraph.

3.1.2) POS tagging

The list which is returned by the word_tokenize function is further passed to the pos_tag function. This function recognizes the parts of speech of each token which it belongs to. All the parts of speech tags that are present in nltk are given below.

Tag	Description	Example	Tag	Description	Example
CC	Coordin. Conjunction	*and, but, or*	SYM	Symbol	*+,%, &*
CD	Cardinal number	*one, two, three*	TO	"to"	*to*
DT	Determiner	*a, the*	UH	Interjection	*ah, oops*
EX	Existential 'there'	*there*	VB	Verb, base form	*eat*
FW	Foreign word	*mea culpa*	VBD	Verb, past tense	*ate*
IN	Preposition/sub-conj	*of, in, by*	VBG	Verb, gerund	*eating*
JJ	Adjective	*yellow*	VBN	Verb, past participle	*eaten*
JJR	Adj., comparative	*bigger*	VBP	Verb, non-3sg pres	*eat*
JJS	Adj., superlative	*wildest*	VBZ	Verb, 3sg pres	*eats*
LS	List item marker	*1, 2, One*	WDT	Wh-determiner	*which, that*
MD	Modal	*can, should*	WP	Wh-pronoun	*what, who*
NN	Noun, sing. or mass	*llama*	WP$	Possessive wh-	*whose*
NNS	Noun, plural	*llamas*	WRB	Wh-adverb	*how, where*
NNP	Proper noun, singular	*IBM*	$	Dollar sign	$
NNPS	Proper noun, plural	*Carolinas*	#	Pound sign	#
PDT	Predeterminer	*all, both*	"	Left quote	(' or ")
POS	Possessive ending	*'s*	"	Right quote	(' or ")
PRP	Personal pronoun	*I, you, he*	(	Left parenthesis	([, (, {, <)
PRP$	Possessive pronoun	*your, one's*	)	Right parenthesis	(],), }, >)
RB	Adverb	*quickly, never*	,	Comma	,
RBR	Adverb, comparative	*faster*	.	Sentence-final punc	(. ! ?)
RBS	Adverb, superlative	*fastest*	:	Mid-sentence punc	(: ; ... – -)
RP	Particle	*up, off*			

3.1.3) ne_chunk

The list of tokens which are appended with their parts of speech are passed to the ne_chunk function. Since all the entities are proper nouns like person

names, locations, geo political entities and monetary values. We only consider the tokens which correspond to NNP tag after returned from ne_chunk function.

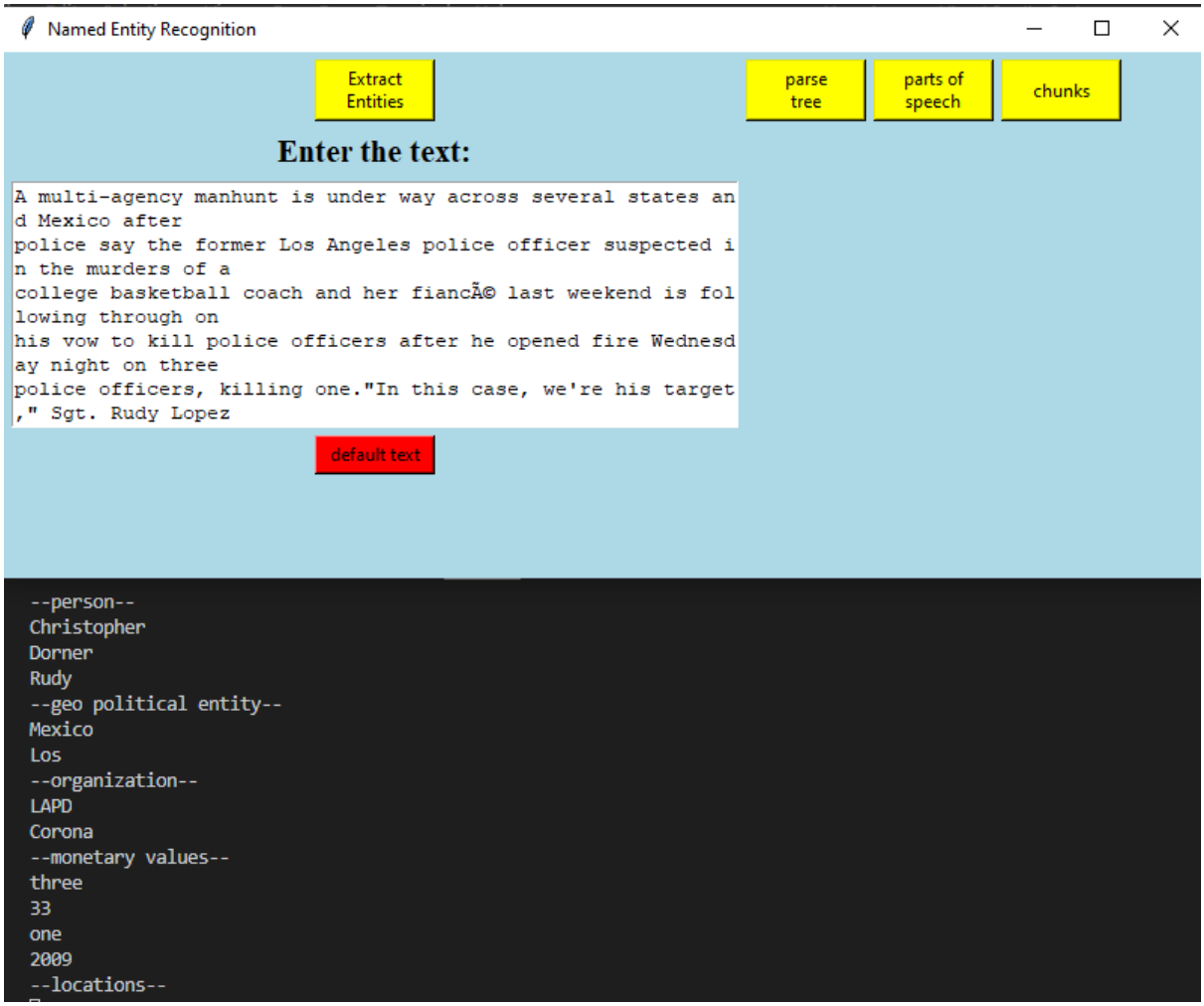
```
been/VBN
identified/VBN
as/IN
(PERSON Christopher/NNP Jordan/NNP Dorner/NNP)
,/,
33/CD
,/,
and/CC
he/PRP
is/VBZ
considered/VBN
extremely/RB
dangerous/JJ
and/CC
armed/VBN
with/IN
multiple/JJ
weapons/NNS
,/,
authorities/NNS
say/VBP
./.
The/DT
killings/NNS
appear/VBP
to/TO
be/VB
```

3.1.4) Extracting entities

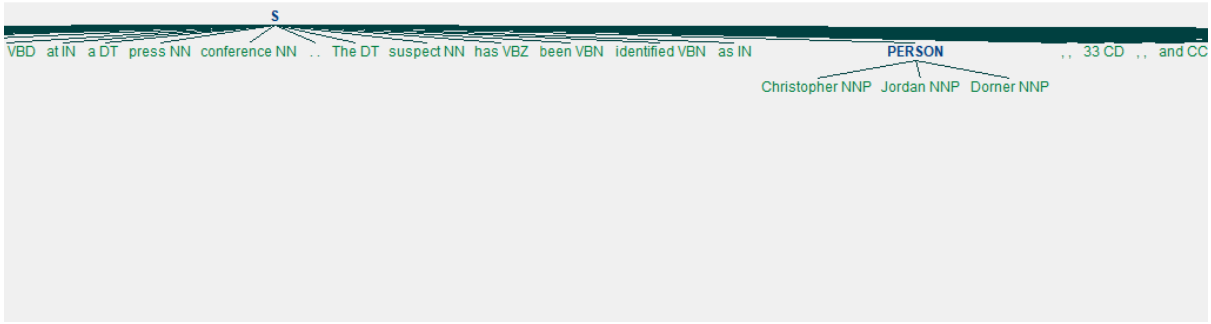
From the string generated by the ne_chunks function, the entities containing similar tags are combined together. Example: (person Christoper/NNP).

The default text button enters text in the textbox. The entities are compared to the string returned by the ne_chunk function. By traversing the string and comparing them to each entity. As ambiguity couldn't be avoided some of the entities could be mis-defined. For example the name Washington could be a person or a state according to the context it is used.

Monetary values like numbers and percentages are tagged with 'CD', persons are tagged with 'PERSON', countries and places are tagged with 'GPE'(geo political entity).

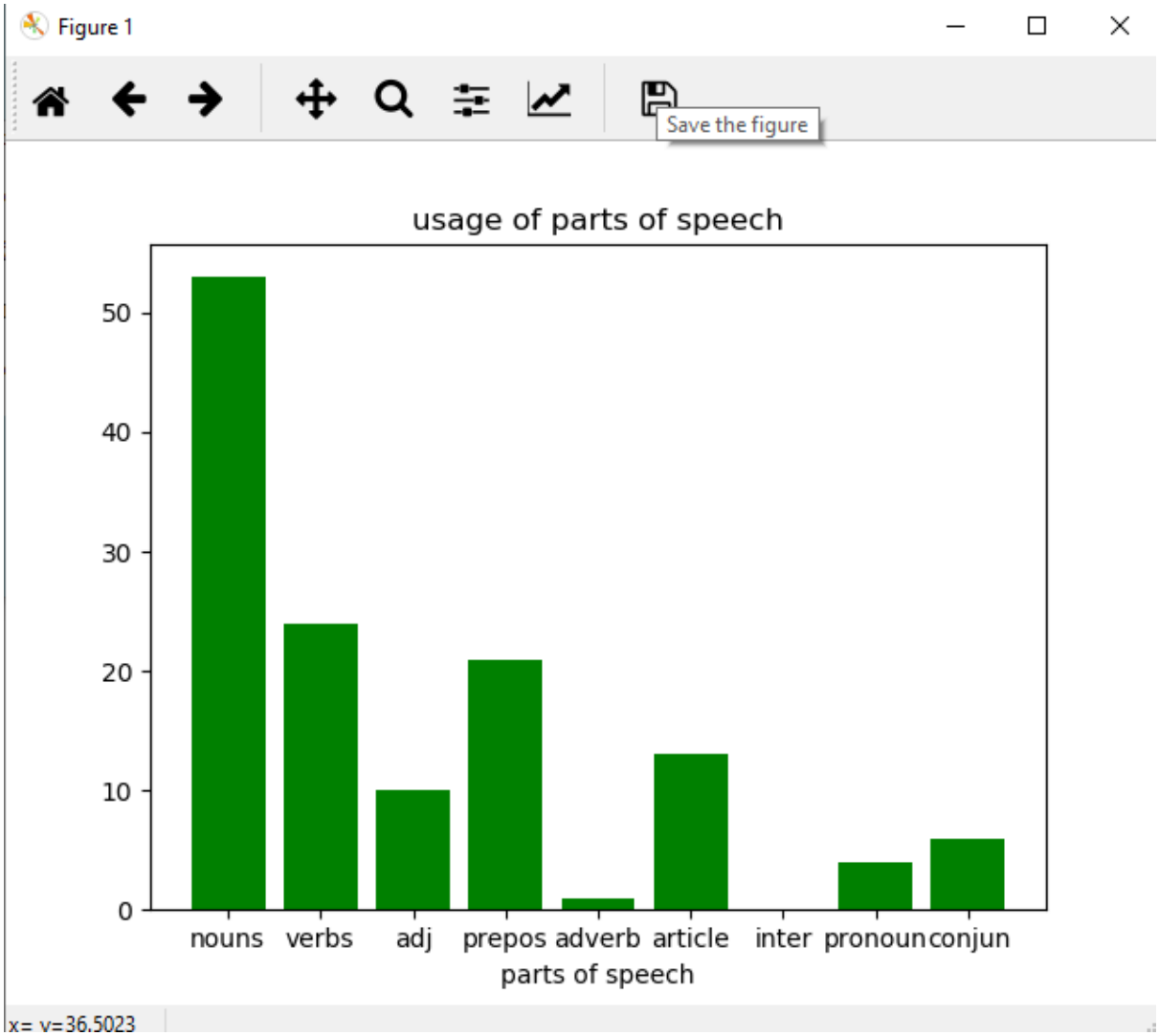


3.1.5) Parse tree



A **parse tree** or **parsing tree** or **derivation tree** or **concrete syntax tree** is an ordered, rooted tree that represents the syntactic structure of a string according to some context-free grammar.

3.1.)Parts of speech graph



4. Work Distribution

Uppala manoj(11810921,roll.no: 47): entity recognition, tkinter (interactive program)

Charapalli tharun(11803413,roll.no: 46): generating a parse tree.

Yarakaraju ravi varma(11802423,roll.no: 40): parts of speech graph

Asireddy Vivekananda reddy(11802339,roll.no: 39): displaying chunks on terminal.

5. Technologies and framework used:

Platform used to implement the code: VSCode
Language used to develop the modules: Python
Libraries used to develop the project: nltk,tkinter,matplotlib
Technology used: Natural language processing

6. **SWOT Analysis:**

1. **Strength:**

It can extract named entities base on their parts of speech tags and chunking results.

2. **Weakness:**

Ambiguity during recognization of entities may occur like Washington can be recognized as a location and also a person based on the context used.

3. **Opportunities:**

Important conclusions can be made by extracting entities, sentiment analysis can be made and chat bots can be built.

4. **Threats:**

There are no threats due to named entity recognition.

7.)

Use Cases of NER Models

Named Entity Recognition has a wide range of applications in the field of Natural Language Processing and Information Retrieval. Few such examples have been listed below :

Automatically Summarizing Resumes :

One of the key challenges faced by the HR Department across companies is to evaluate a gigantic pile of resumes to shortlist candidates. To add to their burden, resumes of applicants are often excessively populated in detail, of which, most of the information is irrelevant to what the evaluator is seeking. With the aim of simplifying this process, through our NER model, we could facilitate evaluation of resumes at a quick glance, thereby simplifying the effort required in shortlisting candidates among a pile of resumes.

Optimizing Search Engine Algorithms :

To design a search engine algorithm, instead of searching for an entered query across the millions of articles and websites online, a more efficient approach would be to run an NER model on the articles once and store the entities associated with them permanently. The key tags in the search query can then be compared with the tags associated with the website articles for a quick and efficient search.

Powering Recommender Systems :

NER can be used in developing algorithms for recommender systems which automatically filter relevant content we might be interested in and accordingly guide us to discover related and unvisited relevant contents based on our previous behaviour. This may be achieved by extracting the entities associated with the content in our history or previous activity and comparing them with label assigned to other unseen content to filter relevant ones.

Simplifying Customer Support :

NER can be used in recognizing relevant entities in customer complaints and feedback such as Product specifications, department or company branch details, so that the feedback is classified accordingly and forwarded to the appropriate department responsible for the identified product.