Match Harry Potter Spells With Their Definition Documentation

Release 1.0

James Brill

Apr 17, 2017

CONTENTS:

CHAPTER

ONE

ABOUT

This Python package has been made as a tool which provides the user with the ability to perform an analysis on two popular vector space models: Google's Word2Vec and Stanford's GloVe. It can either be imported so that other developers can adapt the implemented functions for their own projects, or alternatively, they can run the file hp_spells.py, see the usage notes for more details.

This documentation briefly describes the main features of the program, as well as notes on how to use the program, from installing the necessary dependencies to the parameters that can be passed.

CHAPTER

TWO

CURRENT FEATURES

- Can use either Word2Vec or GloVe to generate Harry Potter spell names.
- A different list of spells can easily be used by replacing the contents of the file “spells.csv” with different contents provided the new content is in the same format.
- Provides evaluation against four different metrics:
    1. Originality
    2. Cosine Similarity
    3. Synoynms
    4. Gibberish Words
- Displays the metrics graphically using violin plots and time series plots.

CHAPTER

THREE

USAGE NOTES

- To install the dependencies, navigate to the base level of the directory and use the command "pip install -r requirements.txt".
- To execute the program navigate to the "hp_spells" directory, and use the command "Python hp_spells.py" to execute the program with default settings.
- There are five different parameters that can be passed to the program.
    1. –help : This displays the help screen which lists what parameters are available.
    2. –glove : When this parameter is passed, the GloVe vector set is loaded instead of Word2Vec.
    3. –exp EXP : An integer supplied to the program which determines how many times the experiment is repeated.
    4. –comp : This flag is used to run the program in comparison mode.
    5. –verbose : When this parameter is passed, the new spell name is printed out on the screen.
- When passing a parameter to the parameter, ensure to use two hyphens.

Harry Potter Spell Generator Functions

hp_spells.**calcProb**(*data*)

Calculates the probabilities for spells of each type.

Parameters **data** (*[[[str, str], int]..]*) – List of spell types and origin language with frequency.

Returns A list of type of spells and their associated probabilities.

hp_spells.**checkStoredWords**(*kwords*, *word*)

This function updates a list of known words with a new word. If the spell type and

language exists in the list the value is append by 1 otherwise, it is appended to the end of the list with a value of 1.

Parameters

- **kwords** (*[[[str, str], int]..]*) – List of spell types and language with associated frequencies.
- **word** (*str*) – One being the spell type and the other being the origin language.

Returns the updated list of known words.

hp_spells.**count_instances**(*fname*)

Reads supplied file, where it splits it up. Then it appends each word to the data

set building a list of words and frequencies using checkStoredWords(kwords, word).

Parameters **fname** (*str*) – This is the name of the CSV file in which the spell data is stored.

Returns returns a list of languages and the probabilities for each one.

hp_spells.**f**(*str*)

Returns the first two chacters from the string.

Parameters **str** (*str*) – A word that is passed.

Returns a string that only contains the first two letters.

hp_spells.**generateScale**(*data*)

This stacks the probabilities of spells so that each spell has a boundary in which

it a spell can be selected over another.

Parameters **data** (*[[[str,str],int,float]..]*) – list of spell names and their associated frequencies and probabilities.

Returns a list of spells and the value between 0-1 in which that name will be selected.

hp_spells.**generateSpell**(*sentence*, *model*, *oword*)

Generates a Spell from a sentence.

Parameters

- **sentence** (*str*) – string which is the definition of the spell you want to create.
- **model** – loaded vector orepresentation of words. :type model: data file loaded.

Returns list containing the spell and the spell type.

hp_spells.**getSpellType**(*scale*, *rndNum*)

Selects a spell according to the random number passed.

Parameters

- **scale** (*[(str,str,float).]*) – A list of tuples which contains the probability associated with each spell and type.
- **rndNum** (*float*) – The random number used to select a spell type.

Returns A string which is the spell type.

hp_spells.**is_synonym**(*n_word*, *o_word*)

This function uses a combination of NLTK's wordnet to list all synonyms for a word and to check if a new word is a synonym.

Parameters

- **n_word** (*str*) – The new word generated.
- **o_word** (*str*) – The original word in the definition.

Returns Returns a boolean indicating whether n_word is a synonym of o_word.

hp_spells.**is_valid**(*string*)

check to see whether a word consists of alpha characters.

Parameters **string** (*str*) – The string to be checked.

Returns Boolean value.

hp_spells.**langCode**(*language*)
Converts a language name into a language code for the translator.

Parameters **language** (*tr*) – Full name of the language, for example latin.

Returns The string code for the language.

hp_spells.**load_vectors**(*path*, *is_binary*)
This loads the vectors supplied by the path.

Parameters

- **path** (*str*) – The path to the vector file
- **is_binary** (*boolean*) – states whether file is a binary file.

Returns The loaded model.

hp_spells.**pigLatin**(*source*)

Takes a source string and converts it from english to pig latin.

Parameters **source** (*str*) – Takes string of english words and changes it into pig latin.

Returns a string containing pig latin words.

hp_spells.**run_experiment**(*model*, *num_experiments*)
This function runs the experiments with the paramters set. It then returns all the necessary data for processing and output.

Parameters

- **model** (*The loaded vector object*) – The vectors loaded.
- **num_experiments** (*int*) – The number of experiments to run.

Returns A list of averages scores, one entry per experiment.

Returns A list of the average number of synonyms produced, one entry per experiment.

Returns The average score across the experiments.

Returns A list of average cosine similarity scores, one entry per experiment.

Returns The number of experiments.

Returns A list containing the number of bogus words produced, one entry per expeirment.

Returns A list containing lists with each sublist containing the scores produced for that definition length.

Returns A list containing list with each sublist containing number of bogus words produced for that definition length.

hp_spells.**sentenceToWord**(*sentence*, *model*, *oword*)
Takes a string and converts it into a vector. Then from that it picks a similar word that doesn't contain an underscore.

Parameters **sentence** (*str*) – A string which contains a sentence to be converted into one word.

Returns A string containing a similar word.

hp_spells.**totalSpells**(*data*)

Counts the number of spells in the dataset.

Parameters **data** (*[[[str, str], int]..]*) – List of spell types and origin language with frequency.

Returns an integer value of total number of spells.

hp_spells.**translate2**(*word*, *lang*)

Translates a word to a target language.

Parameters

- **word** (*str*) – The word you want to convert.
- **lang** (*str*) – the lang code of the language you want to convert to.

Returns a string containing the translated word in the latin alphabet.

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

h

INDEX