International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-4, April 2019

Automatic Paper Drafting Bot

D. Prabhu, S. P. Maniraj, Aakanksha Thakur, M. Bhagyashree, Richa Gupta

***Abstract:** Research plays a significant role in the development of any field. To project new ideas, many enthusiast publish uncountable papers every year. Time and energy spent by researchers on writing the paper is huge. Because writing research paper also bring rules, an automated system can be designed to follow the rules so that human labour can be utilised in other important tasks like analysis and research of the project. Text in the document are categorized using SVM - Support Vector Machine text classification algorithm. In this paper, an automated paper editing bot is presented which edits and drafts the paper based on user voice commands. This reduces handwork of the researcher up to a great degree. The bot is developed by using basic low weight Python libraries Py Auto GUI and Speech Recognition.*

***Keywords:** Py Auto GUI, Speech Recognition, auto paper editing.*

I. INTRODUCTION

Advancements in technologies and new project ideas evolve every year exponentially. This is necessary to establish a system that overcomes the existing drawbacks and continually grow over time. In order to obtain this, many enthusiast invest their priced energy and time to create new ideas that function better than the living system. Research papers play critical job in the demonstration of these ideas. Studies suggest that majority of research time is spent on drafting the paper. Drafting of paper doesn't require human intelligence and can be piled upon automated machines. Many Technology innovations like speech to text, machine learning and big data have already reduced the heavy load from user shoulders up to a great degree. A good combination of these libraries can be employed to develop automated paper bots. Any research includes analysis of Idea and paper drafting. Former task requires human Intelligence and hence is highly difficult for machines than humans the latter task requires labour work based on set of rules and hence is easier and faster for machines than it is for humans.

II. RELATED WORK

Voice recognition automates the task upto a great level. Speech to text is now very prevalent in many applications. [1] describes about a voice directed self-regulating system for homes. It operates with voice recognition system developed by google.

Manuscript published on 30 April 2019.
***** Correspondence Author (s)
D. Prabhu, Assistant professor, SRM Institute of science & Technology, Chennai
S. P. Maniraj, Assistant professor, SRM Institute of science & Technology, Chennai
Aakanksha Thakur, UG Scholars, SRM Institute of science & Technology, Chennai
M. Bhagyashree, UG Scholars, SRM Institute of science & Technology, Chennai
Richa Gupta, UG Scholars, SRM Institute of science & Technology, Chennai

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open access article under the CC-BY-NC-ND license http://creativecommons.org/licenses/by-nc-nd/4.0/

It bears comprehensive group of libraries that help in easy construction of complex projects. Most of these APIs reside on cloud which making it accessible on almost every platform. Certain APIs are free while the rest charge minimal fee. Python packages that concentrate on speech recognition reside on PyPI [6]. Most commonly known instances of this package are Google-cloud-speech and Speech Recognition. Cloud speech API is paid and asks for a valid google account. It includes multiple steps of registration queries like information related to name of the project [2]. In order to expand more user friendly systems many researches are done on the standard actions performed by humans. Similar system is developed in [4] where the computer program interacts with elderly people and interact with them similar to human to prevent them from dementia. It works in accordance with natural language processing to differentiate among multiple accents in the speech. Speech recognition can be used in many other fields to perform wide range of operations. [8] presents the usage of voice recognition system in call centers. The system develops many speech based subsystems like dialog management and speech synthesis which forms the foundation of any call center. Other than recognizing speech many humanoid robots with image recognition abilities were also developed. OpenCV is a python library that can do this job efficiently. OpenCV is particularly a software which is used especially for image processing in real time [3]. Text classification is an activity of classifying the text and assigning tags to them. It is a considerable action in automated systems concerned with human languages [7]. This field has been finding its place since the evolution of data computation. [5] describes the most widely accepted classification algorithms in particular naive bayes classifier, nearest neighbour and decision tree classifier. [11] presents short text classification algorithms. Short text refer to the length of the shorter text content. Text messages fall under this domain. [9] outlines a brief picture of SVM based text classification algorithm. Deep analysis of performance comparison on two standard dataset: Reuter-21578 and TICP-AP was executed. Each of them contain newswire data.

III. DESIGN OF BOT

Automatic paper editor is outlined using low weight Python libraries. Python is a highly flexible language and provides distinct flavour of libraries for automation test that performs basic user actions like typing and selection of text programmatically based on users voice commands. It is designed using Python 2.7. Design includes three major modules.

A. PyAutoGUI

PyAutoGUI is a multiplatform GUI self-operating module that works on Python 2 and 3. This library provides the control of mouse and keyboard of the executing system programmatically.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)
© Copyright: All rights reserved.

Programs can be developed to automate real time GUI task using this library. PyAutoGUI can simulate operations to control motion, choice selection and dragging activities of the mouse. It also works with keyboard functions like pressing keys, pressing and holding keys, and pressing keyboard hotkey combinations. Hotkey combinations can include shortcut key combinations like ctrl+a for select all and ctrl+s for save. PyAutoGUI can be implemented on different platforms. On Windows, PyAutoGUI does not require any dependencies like pywin32 module since it uses python's own ctypes module. Whereas on OS X, PyAutoGUI necessitate PyObjC installed for the AppKit and Quartz modules. Pybjc and pyobjc-core modules are present in PyPI. Linux requires different modules for different version of python. Python 2 supports python-xlib while python 3 supports python3-Xlib.

On top of already existing advantages of PyAutoGUI, one of the best feature it supports is FAILSAFE. This feature helps the program from going out of control. Certain logical miscalculations may lead the program constantly loop over mouse or keyboard clicks preventing the user from further access. In practice FAILSAFE feature is active by default and can be deactivated by setting the value to false. PyAutoGUI also supports image related functions like capturing screenshots. Although to deal with image related data it uses Pillow/PIL library. Other than capturing screenshots, it can also locate an image on the screen and return the position of the image in terms of coordinates. PyAutoGUI assist message box functions. These operations are applied to exhibit alerts and provide support to the user. Message box may also include ok and cancel buttons depending on the purpose of usage.

B. SpeechRecognition

Speech recognition library of python permits program application to be voice directed. Modern speech recognition structure depends on Hidden Markov Model (HMM). In numerous present-day speech recognition systems, neural networks are used to rationalize the speech signal using feature transformation and dimensionality reduction expertise prior to HMM identification. Voice task detectors are used to eliminate chunks of the audio that are not speech signal. This averts the recognizer from spending time analyzing portions of the signal that are not needed. PyPI supports different types of speech to text libraries[6-7].

- apiai
- wit
- assemblyai
- Houndify API
- google-cloud-speech
- pocketsphinx
- SpeechRecognition
- watson-developer-cloud
- Snowboy Hotword Detection

The bot is made easy to use by embedding speech to text. Speech recognition allows visually and physically damaged to interact with sophisticated commodities and services without hefty GUI interface. Other than speech recognition python also supports above listed speech to text package, many of which are online based.

Recognizer class forms the foundation of SpeechRecognition. Object created from this class carry wide range of functionality and settings to identify speech from root audio. Instance of this class hold disparate methods for different APIs. For instance, google cloud speech uses recognize_google_cloud() method for recognizing speech. SpeechRecognition passes the audio file where speech is converted to text. It supports specific file types out of which the most commonly used is WAV format. WAV is a typical audio format mainly supported on windows systems. Many other formats are reinforced on additional systems.

This project uses speech recognition version 3.8.1. Detection of speech demands audio data. Speech recognition library makes this job easier by using built-in scripts to analyse audio input from microphone. Hence, no heavy code is needed to make the bot voice enabled. This package acts as a wrapper for many speech API and is thus highly flexible. The bot discussed in this paper employs speech recognition library by google along with the standard speech API. A microphone object from the recognizer class is created which records the audio. This audio is passed through Google recognizer API which gives text output.

C. Text classification and analysis

Text organization and classification is an activity of allocating tags or categories to text according to its material text. It is the underlying operation in language processing with extensive applications such as emotion analysis, topic tagging, junk content recognition, and target detection[9]. In general, classifier takes text as input and categorized text as output. Depending on the requirement, text categorization can be either automatic or manual. Manual text organization system is called rule based system. This system is transparent to user and can be upgraded later. The major pitfall of this system is, it is very gradual for complex content analysis and tagging. Automated systems overcome this drawback. Numerous intelligent algorithms have been crafted to do perform content classification task efficiently.

The initial move with regards to coaching a classifier with machine learning is property withdrawal. This technique is used to modify each text data into a quantitative rendition in a vector fashion. Bag of words approach is the most routinely used operation for feature extraction, where a vector depicts the number of occurence of a word in a preset dictionary of words. Later, the expert system algorithm is provided with data that consists of sets of feature pairs and tags to create a classification model. After enough training with the selected data, the model can be used for testing and making accurate predictions. Similar characteristic uprooting is used to convert unknown text to feature sets which can be placed into the classification model to get predictions on tags. Text organization with machine learning is customarily much more precise than traditional rule based systems, chiefly on complex categorization tasks. Besides, classifiers developed with these expert systems are not difficult to maintain and one can regularly tag new examples to learn new tasks. Multiple text classification algorithms have been proposed over the years. Following are the most commonly used.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)
© Copyright: All rights reserved.

Naive Bayes

Naive Bayes is a demographic algorithm majorly used for text organization. Building quality rich results even with insufficient data makes this algorithm more dominant. This algorithm works on bayes theorem. Bayes theorem calculates the prospect of an occurrence computed from preceding data knowledge. In text classification scenario, any text content that needs to be classified the category is computed based on the frequency of the text or related texts in a certain tag. Tag with maximum frequency of the text would later be assigned to the content.

Support Vector Machines (SVM)

Support vector machines is a machine learning algorithm that finds its place in text classification very significantly. In text classification this algorithm classifies text based on division made by the hyperplane in a coordinate system. Either sides of the plane group into one specific category. Hyperplane must be detected precisely to segregate multiple tags. Coordinate system includes the plot of textual data items separated into categories. Distance of these data items from the hyperplane is called margin. A system may include multiple hyperplanes and correct hyperplane must be selected based on the marginal values of each hyperplane. Hyperplane with high margin values prove to be better in classification due to its firmness.

SVM can be executed in python using scikit-learn library. Python scikit-learn is employed to execute numerous machine learning algorithms. This project uses scikit-learn to implement SVM algorithm required for text analysis. Additionally scikit-learn also supports assisting functions for Naive bayes algorithms. Text from speech to text module is passed onto the text analysis function where the loaded text are classified into groups. Based on the user's wish particular operations are triggered. Subsequently, PyAutoGUI module executes the prefered operation.

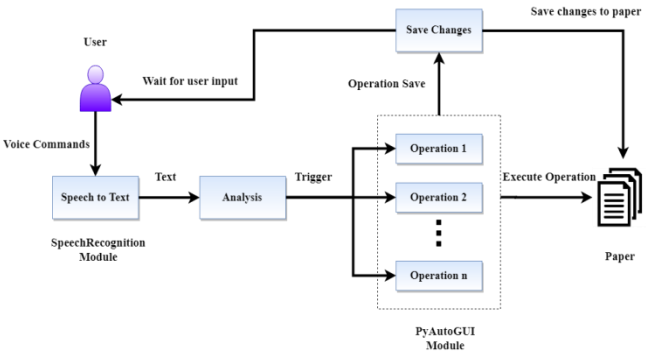

Figure 1. Architecture of the proposed model

IV. IMPLEMENTATION

The project is implemented using Python 2.7 along with the libraries mentioned in the preceding modules. User uses the interactive Python bot on one part of the screen and the paper template or the paper to be edited spreads out on the other part of the screen. User submits voice commands to the program for a specific job to be done. Program converts the speech to text using speech recognition library. Then the obtained text command is analysed and specific operation module is triggered. Triggered module performs the job by following editing sequences using PyAutoGUI. Following the effective execution of the function or operation module, the changes are reflected on the document present on the screen. Paper includes different sections based on the content. Hence, abstract text cannot be written in conclusion area and vice versa. This project has divided the paper into two categories. First category is the abstract. As abstract summarizes the whole paper, it portrays a significant role. Every paper contains only single abstract. Thus, abstract must be edited and not added. Second category is the module and unlike abstract modules can be in any number. Modules may be related to any subject matter containing multiple paragraphs while abstract is written only in one paragraph. Almost every research paper follows these basic rules. This project is designed based on these conditions to make paper writing more user friendly. User is constantly asked for operations to be performed on the paper until user desires to exit the process.

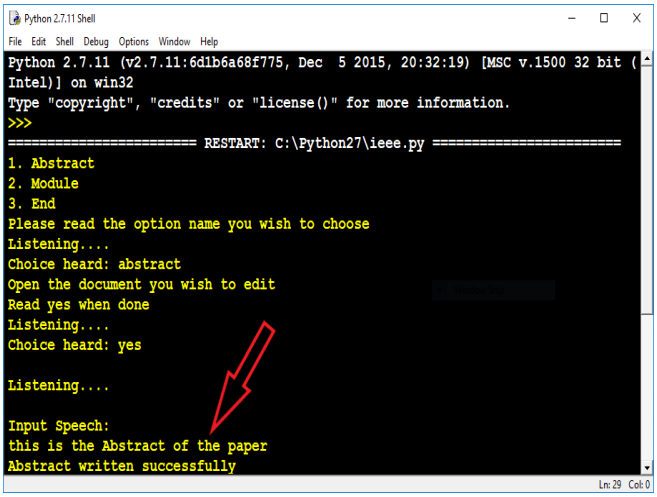

Figure 2. Abstract writing through voice controlled bot using python shell

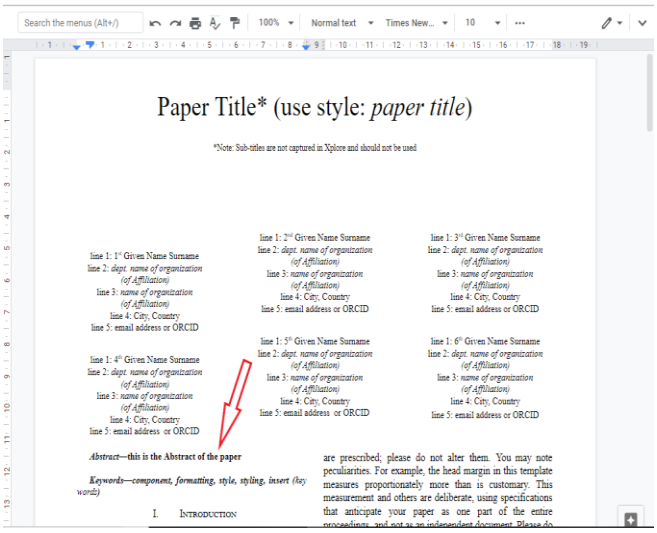

Figure 3. Changes in the abstract are reflected in the document

Editing of paper in this project is done on the standard template available online. Drafting bot also works well with incomplete paper that needs further writing and can help in adding or removing modules.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)
© Copyright: All rights reserved.

Figure 2. displays the interaction of user with the program using python shell. It is clear from the image that user desired to edit the abstract was prompted to read out the abstract. This speech was recorded by the system and conversion from speech to text occurred through google speech to text. Text is then categorized as abstract portion of the document in text analysis operation. Finally using PyAutoGUI API abstract is written on the desired section of the document. Above image illustrates the modifications made in the paper as a result of the operations selected by the user displayed in Figure 2. User when selects a specific option displayed in the bot, modules related to that specific option are triggered. For instance, abstract option triggers abstract module which after text analysis invokes the speech recognizer module from the recognizer class. This creates microphone object of the recognizer class that does the required listening.

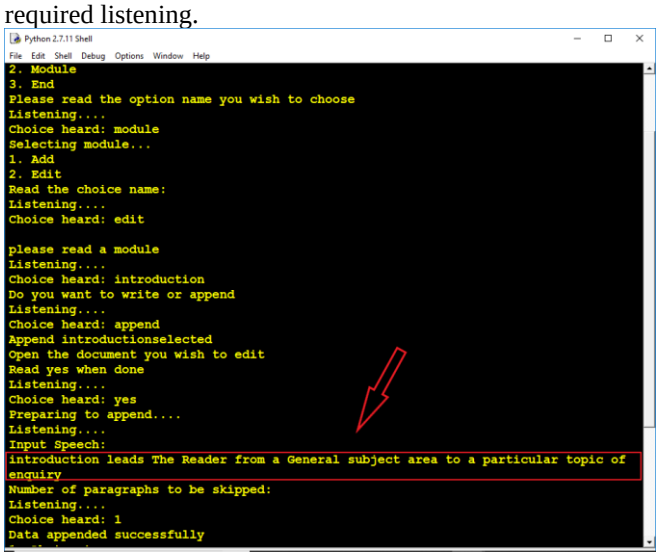

Figure 4. Module writing through voice controlled bot using python shell

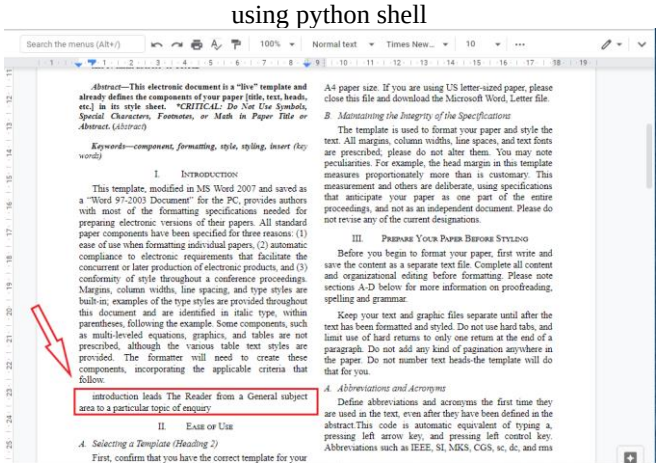

Figure 5. Changes in the introduction paragraph are reflected in the document

Unlike abstract modules can be in any number. Hence, data about modules must be stored and the changes must be saved. Interaction of the user with the system where the user attempts to edit introduction section of the paper is shown in the image 4. A module can either be created or changes can be made on an already existing module. Write option of the system assists user to write a section of the paper freshly i.e by deleting old text content. Append alternative can be helpful to catch up from where the user left off. This option does not necessarily deletes the content but continues from the old content. The system also gives a choice to append the text content in between the paragraphs. It prompts the number of paragraphs user desires to skip in a particular module as shown in the paragraph. Observe that the user chose one hence one paragraph was skipped and the text content was placed in the desired location. As specified, introduction paragraph spoken by the user is placed on the right place. This change is shown in figure 5.

V. CONCLUSION

As the matter of fact, this project has an idea of minimizing the time spent and handwork involved while drafting the paper. An automated paper drafting bot is developed using prebuilt python libraries and text classification algorithms. Libraries used in the project enhance user interaction with the bot and makes it easy to use. This project uses SpeechRecognition and PyAutoGUI APIs which provides the advantage of high quality user experience. Text classification algorithms provide the text analysis required in order to ease the editing operations. Use of real time software tools makes this project highly scalable and compatible for wide range of users.

VI. FUTURE WORK

Current AI systems like Alexa, Siri and google assistant perform numerous general operations. Paper drafting bot can be built on top of an existing AI agent in order to increase the skill set of the existing systems. Current system concentrates on the prevalent sections of editing and details like font and style editing can be proposed in the future model of this project. Secure access of the document and user authentication can be advanced to make the system more reliable and robust. Methodical algorithms can be manoeuvred to expand the potential of the system.

REFERENCES

1. Prashanth Kannan, Saai Krishnan Udaykumar, K.Ruwaid Ahmed, “Automation Using Voice Recognition with Python SL4A Script for Android Devices”, 2014 International Conference on Industrial Automation, Information and Communications Technology (IAICT).
2. Septimiu Mischie, Liliana Mâiu-Iovan, Gabriel Gpresc, “Implementation of Google Assistant on Rasberry Pi”, 2018 International Symposium on Electronics and Telecommunications (ISETC).
3. U. Bharath Sai, K. Sivanagamani, B Satish, M Ranga Rao, “Voice controlled Humanoid Robot with artificial vision”, 2017 International Conference on Trends in Electronics and Informatics (ICEI).
4. Wei-De Liu, Kai-Yuan Chuang, Kuo-Yi Chen, “The Design and implementation of a chatbot's character for elderly care”, 2018 International Conference on System Science and Engineering (ICSSE).
5. Y. H. Li and A. K. Jain, “Classification of Text Documents”, 1998, Vol 41, No. 8, The Computer Journal.
6. https://realpython.com/python-speech-recognition/
7. https://pypi.org/project/SpeechRecognition
8. Marian Ceaparu, Stefan-Adrian Toma, Svetlana Segărceanu, Inge Gavăt, “Voice-based User Interaction System for Call-centers, Using a Small Vocabulary for Romanian”, 2018 International Conference on Communications (COMM).
9. Zi-qiang Wang, Xia Sun, De-Xian Zhang, Xin Li, “An Optimal SVM-Based Text Classification Algorithm”, Fifth International Conference on Machine Learning and Cybernetics, Dalian, 13-16 August 2006.
10. David M. Beazley, Python Essential Reference, 4th edition, AddisonWesley Professional, Jun 29, 2009.
11. Zhou Faguo, Zhang Fan, Yang Bingru, Yu Xingang, “Research on Short Text Classification Algorithm Based on Statistics and Rules”, 2010 Third International Symposium on Electronic Commerce and Security

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)
© Copyright: All rights reserved.