

# IJIRE-0000755\_T

*by* FDRP Journals

---

**Submission date:** 26-Jun-2024 09:30PM (UTC-0700)

**Submission ID:** 2393463869

**File name:** IJIRE-0000755.docx (281.31K)

**Word count:** 3130

**Character count:** 18931

# An Air Gesture Based Keyboard for Visually Impaired People Using Machine Learning And Embedded System Approach

<sup>1</sup> Hemashree M  
Dept. of ECE, BGS Institute of Technology  
Adichunchanagiri University  
B G Nagara, India  
[bmhemashree29@gmail.com](mailto:bmhemashree29@gmail.com)

<sup>2</sup> Kavitha B C  
Dept. of ECE, BGS Institute of Technology  
Adichunchanagiri University  
B G Nagara, India  
[kavithabc@bgsit.ac.in](mailto:kavithabc@bgsit.ac.in)

<sup>3</sup> Leena Ranganath  
Dept. of ECE, BGS Institute of Technology  
Adichunchanagiri University  
B G Nagara, India  
[ranganathleena@gmail.com](mailto:ranganathleena@gmail.com)

<sup>4</sup> Suhas  
Dept. of ECE, BGS Institute of Technology  
Adichunchanagiri University  
B G Nagara, India  
[suhasbhupal243@gmail.com](mailto:suhasbhupal243@gmail.com)

**Abstract**— This report outlines the development of an air gesture keyboard utilizing machine learning technology, designed to assist visually impaired individuals in entering text more easily. This innovative system removes the need for physical keyboards, instead employing a remote-based system that translates hand gestures into text through the use of an accelerometer, Arduino, and switches. There is a significant need for alternative input methods for visually impaired individuals due to the limitations of existing technologies. The proposed air gesture keyboard captures and interprets hand motions in the air, enabling users to input text without physical contact. A literature review was conducted to examine related research on hand gesture recognition using various techniques.

**Keywords**— Machine learning, gesture, Microcontroller, visually impaired.

## I. INTRODUCTION

The keyboard has been an essential component of computer systems, allowing users to input data through key presses. While physical keyboards are still widely used, virtual keyboards have become popular with the rise of mobile devices. However, speech-to-text technology often lacks accuracy. To address this issue, remote-based gesture keyboards have been developed, utilizing machine learning algorithms and Python programming.

These keyboards employ accelerometers to capture air gestures and convert them into text, removing the need for a specific layout and enabling multilingual functionality. The system comprises an accelerometer, an Arduino microcontroller, and switches, with the Arduino IDE configured to set the baud rate to 38400. The scikit-learn library is used to convert accelerometer signals into letters, which are stored in a dataset and trained using machine learning algorithms to recognize gestures in various languages.

The remote-based gesture keyboard provides several advantages over traditional keyboards, including improved accessibility for individuals with disabilities, enhanced accuracy, freedom from layout constraints, and increased durability. It can connect to devices wirelessly via Bluetooth or through a wired USB connection. This innovative approach allows users to input text and numbers by moving their hand in specific character motions in the air, simulating the experience of writing with a pen.

## II. LITERATURE SURVEY

Hand gestures serve as a natural and intuitive form of human-computer interaction (HCI), necessitating the development of computer interfaces capable of recognizing them in real time to improve HCI efficiency. One innovative approach, explored by Shakunthala Devi and Revathi, involves real-time hand gesture recognition using an AVR microcontroller. Instead of relying on traditional digital cameras, their system utilizes MEMS accelerometer sensors to measure tilt, shock, and vibration, providing a more mobile, cost-effective, and versatile solution.

In another promising area of research, Jaramillo and Benalcazar focus on real-time hand gesture recognition through electromyography (EMG) combined with machine learning techniques. Despite challenges such as noisy EMG signals and high dimensionality, their model includes stages of signal acquisition, preprocessing, feature extraction, classification, and post-processing to enhance accuracy and gesture recognition capabilities. Advances in this field could potentially benefit other areas such as face and audio recognition.

Additionally, Tai, Jhang, Liao, Teng, and Hwang have developed a sensor-based continuous hand gesture recognition algorithm using long short-term memory (LSTM). This algorithm, requiring only basic accelerometers and gyroscopes, employs a many-to-many LSTM architecture to produce output paths from input sensory data sequences. By performing maximum a posteriori estimation on observed paths, the algorithm achieves robust classification results.

These studies collectively advance the field of hand gesture recognition by introducing innovative methodologies and tackling various challenges. By utilizing AVR microcontrollers, EMG signals, and LSTM algorithms, researchers are working to improve real-time hand gesture recognition, thereby enhancing natural and efficient human-computer interaction.

### III. TECHNICAL Requirements

#### 3.1 Accelerometer Mpu6050



**Fig. 3.1 Accelerometer Mpu6050**

The MPU-6050 is a versatile motion processing device that combines a 3-axis accelerometer and a 3-axis MEMS gyroscope in a single chip. It features a built-in 6-axis motion fusion algorithm-capable digital motion processor, providing comprehensive motion tracking capabilities. Additionally, it includes a temperature sensor for monitoring environmental conditions. A notable feature of the MPU-6050 is its ability to communicate with microcontrollers via an I2C bus interface, simplifying integration into various systems. When a 3-axis magnetometer is connected to the auxiliary I2C bus, the MPU-6050 can produce a complete 9-axis motion fusion output.

#### 3.2 Arduino UNO



**Fig. 3.2 Arduino UNO**

In this project, the microcontroller serves as the central processing unit for all sensor data, playing a vital role. Its main function is to convert raw sensor readings into usable information for the computer system. The microcontroller facilitates a communication conduit, ensuring efficient processing and utilization of the sensor data.

#### 3.3 Button Switch



**Fig. 3.3 Button Switch**

During the design phase, a crucial decision arose concerning how to alert the microprocessor of a sign being made. One potential solution considered was the implementation of a single push button, depicted in Figure 3, which would be pressed to activate the system's prediction functionality.

#### 3.4 Bluetooth Device



**Fig. 3.4 Bluetooth Device**

"A Bluetooth module facilitates the transmission of patterns generated by the MPU6050 to the system and the reception of data from the system."

### IV. PROPOSED IMPLEMENTATION

#### 4.1 Circuit Diagram

The HC-06 module facilitates wireless communication between an Arduino Pro Micro and an accelerometer, establishing a link for data transmission. The Arduino connects to the accelerometer using the SDA and SCL pins, with power supplied through grounding the system. The accelerometer captures motion in the x and y directions. For motion recognition, PyGARL, a Python gesture analysis and recognition library, is utilized, making use of the scikit-learn package. This library enables the storage of text-based data and seamless integration of additional data. Gesture data undergoes processing through support vector machine regression, a machine learning technique. Upon finding a match, the corresponding data is displayed on the screen.

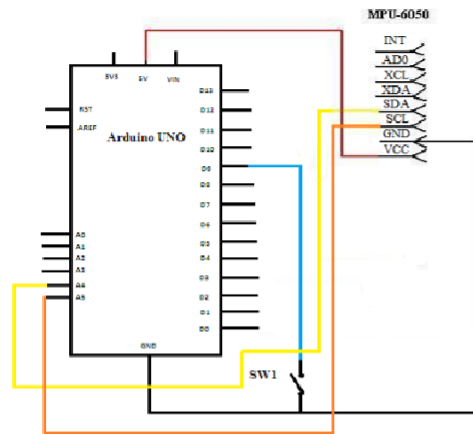


Fig. 4.1 Circuit Diagram of Gesture Keyboard using Arduino

Accurately translating accelerometer data into text presents challenges, particularly in air-writing scenarios where the vocabulary can be extensive compared to standard gesture recognition systems. To address this, a specialized library called Gesture Keyboard was developed. To gather accelerometer data, an Arduino module equipped with an MPU-6050 accelerometer is employed, transmitting the data to a computer. On the computer, a Python package utilizes the Support Vector Machine algorithm from Scikit-learn to categorize signals into letters. This method, implemented in the Gesture Keyboard library, enables precise interpretation of accelerometer data, facilitating accurate translation of motions into text.

#### 4.2 Creation of gesture recognition software

Scikit-learn, a Python programming library focused on machine learning, provides a wide range of algorithms for classification, regression, and clustering tasks, such as support vector machines, random forests, and gradient boosting. It is designed to seamlessly integrate with other Python numerical and scientific libraries. In tasks like gesture recognition or other supervised learning applications, the machine learning process advances through multiple stages before the final classifier produces an efficient output. Each stage operates independently and serves a unique purpose in improving classification accuracy.

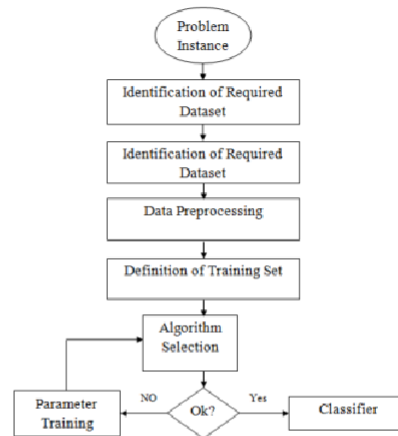


Fig. 4.2 Flow diagram for machine learning algorithm

- **Required Dataset:** The initial step involves defining the data attributes essential for determining the output goal. All values must adhere to specific constraints. The dataset is then prepared to identify various parameters that can be further extracted.
- **Data Pre-Processing:** Data pre-processing varies depending on the specific issue at hand. It includes instance selection, which aids in effectively learning from large datasets while addressing noise management issues. Instance selection optimization aims to reduce sample size while preserving data quality, allowing data classification algorithms to handle large datasets efficiently. There are numerous techniques for selecting examples from a large collection.
- **Training Set:** Logical and constraint-satisfying training inputs are obtained at each input cycle once a well-specified input variable is globally available. The training set consists of similar pieces organized into separate entities. Serving as the starting point for classification space, the training set facilitates experimentation and trend monitoring. It comprises attribute key-value pairs for various instances of the set.

### 4.3 Classification Algorithms for Gesture Recognition

In this section, we will provide a detailed overview of our data collection process and objective values, followed by an assessment of the algorithm's performance in teaching gestures.

**Gesture Dataset:** We will create a new dataset to enhance accuracy, studying the various movements within this dataset for better understanding. Due to the time sequences involved, continuous contribution of training data to the learning set is essential as the precision of the accelerometer decreases with each movement or change of accelerometer. Our system is capable of handling up to 40 different motions, each corresponding to a single English alphabet letter. Pressing the online button initiates the recording of a fresh set of data inputs for gestures, utilizing the following parameters:

- **Target:** Instructs the module to record fresh gesture examples.
- **"a":** Denotes the distinctive gesture with a length of 1.
- **0:** Batch Number; each new batch must have a unique number to prevent overwriting.
- **Port:** Specifies the connection to the Arduino or microcontroller's serial port.

It is recommended to record each sample and save it as a separate file in the data folder for future analysis. Once the dataset is prepared, we select a suitable model to train these inputs. To identify the optimal model, we evaluate two of the most popular classification algorithms using our dataset. Choosing the ideal algorithm can be challenging.

**Classification Algorithms:** These algorithms utilize a group of variables such as "feature variable," "independent variable," or "predictor variable" to generate predictions. The class variable, representing the outcome of the predictions, is associated with the input data and can be categorical or continuous. The choice of the classifier is based on the algorithm's prediction accuracy, evaluated using a variable called the "Confusion matrix." The Confusion matrix assists in measuring the proportion of test data properly labeled by the classifier.

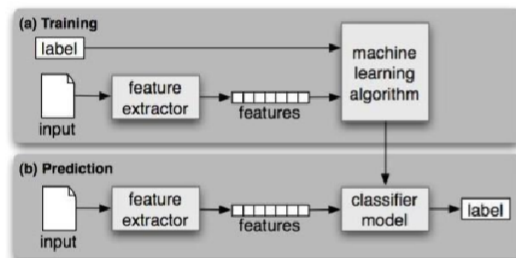


Fig.7 Supervised Classification of data

Support Vector Machines (SVMs) are based on the concept of a "margin," which is a hyperplane that separates two classes of data. The objective is to maximize this margin, positioning the hyperplane at a maximum distance from the instances on each side. Maximizing the margin serves to reduce the upper bound on the predicted generalization error, as demonstrated through mathematical analysis.

SVMs, introduced in 1995, are proficient at classifying data points by drawing lines to separate them into categories. In classification tasks, both training and testing datasets are typically utilized. The training set consists of examples, where each instance includes a "target value" and multiple "attributes." The primary goal of SVMs is to develop a model from the training data that can accurately predict the target values of test data based solely on their attributes.

To address an optimization problem, an SVM requires a training set of instance-label pairs  $(x_i, y_i)$  with values ranging from 1 to 1. Each data instance needs to be represented by a vector of real numbers to utilize this approach. Categorical attributes

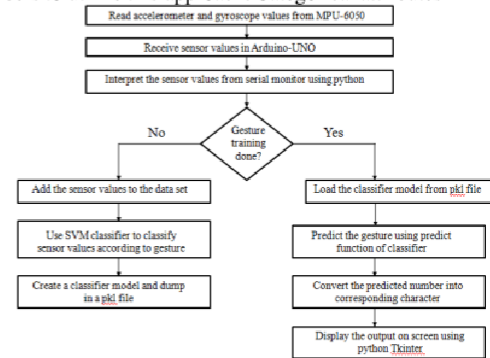


Fig.8 Flow Diagram of Gesture Keyboard.

The system initializes without prior information. It begins by interpreting hardware-based motions captured by the MPU6050 sensor linked to Arduino. Continuously transmitting hand location information, the sensor enables tracking of each gesture's trajectory. Integration of these sensor readings facilitates inference of hand trajectory during gestures.

Subsequently, the collected sensor readings are compiled into a dataset for analysis. During the machine's training phase, users assign specific characters to correspond with sensor values. This mapping allows the computer to identify the intended character from an airborne motion. For example, users may associate hand movements used to form the letter 'a' with its representation on a computer screen. Linking the physical gesture of forming the letter 'a' with its representation on a computer screen.



## V. COLLECTING DATA FROM GESTURES

The MPU6050 accelerometer data is transmitted to a PC. On the PC, a Python library is employed, utilizing the Scikit-learn Support Vector Machine algorithm to classify the signals into letters. Scikit-learn is predominantly developed in Python, with certain core algorithms implemented for performance. Support vector machines are executed using a wrapper around LIBSVM within the Python environment. Figure 9 depicts the Python Machine Learning Script.

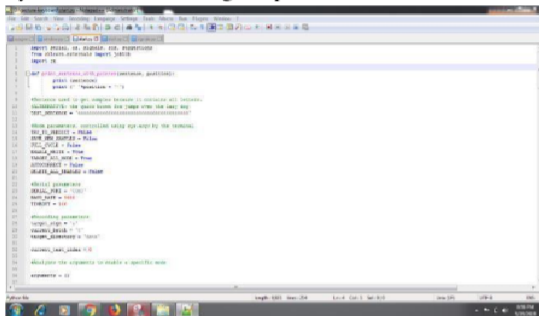


Fig. 9 Python Machine Learning Script

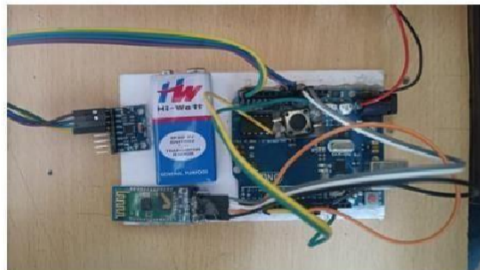


Fig. 10 Implementation of hardware

Figure 10 illustrates the hardware implementation of the air gesture keyboard designed for visually impaired individuals utilizing machine learning technology.

## VI. RESULTS

**Initiating the Python Script:** To start the Python script, execute the following command in the project directory via Command Prompt/Terminal: `python start.py target=a:0`. This command triggers the execution of the "start.py" script with "target" set to 'a'. Figure 11 illustrates the process of starting the Python script using Command Prompt.

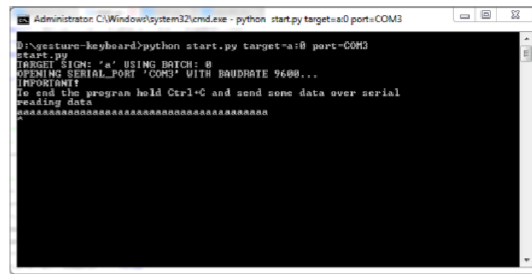


Fig.11 Starting of the Python script using Command prompt

The "target" parameter specifies the specific character for which gesture data is stored. In this scenario, with the target set to 'a', the sensor values are recorded for the character 'a' in a predefined file format.

**Interpreting and storing the data:** Upon initiating the script, the device powers on, and gestures are performed by holding the device and pressing the Start Button. The script interprets values from the Serial monitor and saves them in a dataset file corresponding to the specified target while the button is pressed. Releasing the button stores one set of values. To ensure adequate data for prediction, the button must be pressed and released multiple times to generate multiple sets of data. It was determined that 40 sets were sufficient for our requirements.

**Learning from the data:** The "Learn.py" script handles the learning logic, creating and training a classifier based on data received from the Arduino. The stored dataset is utilized to classify data into categories (based on letters) and sensor data. Categories are derived from file names, while sensor data is extracted from file content. Letters are represented as numbers in the model, as the classifier relies on numerical values instead of characters. X\_Data stores sensor data, and Y\_Data stores category labels. Figure 12 illustrates the learning process using trained data values.

**Support Vector Machine (SVM) Initialization:** A support vector machine is defined for classification using the Scikit-learn module. By creating a model and evaluating it for each set of algorithm parameters listed on a grid, the matrix-look approach is employed for parameter tuning. This technique utilizes cross-validated grid search to exhaustively explore parameter values and optimize the estimator's parameters utilized in these approaches.

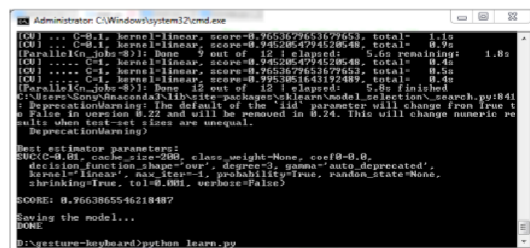
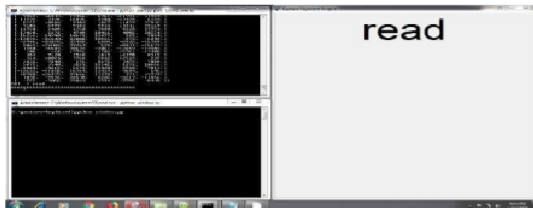


Fig.12 Learning from trained data values

**Executing Predict():** In essence, fitting refers to the training process. Once trained, the model can make predictions, typically accomplished by calling the `predict()` function. In the given equation, this method determines the coefficients. For a classifier, the `predict()` method enables the categorization of incoming data points, such as those from a test set. When applied to incoming data points for regression, the model may extrapolate or interpolate. Fitting essentially involves categorizing sensor values according to their respective categories, such as storing sensor readings associated with the letter "a." The classifier object is serialized and saved into a model. `pkl` file for subsequent use in making predictions.

### 6.1 Predicting the Gesture

After the device is trained for a specific gesture, predictions can be made by running the following command on the computer: ``python start.py predict``. This script loads the classifier from the `model.pkl` file generated during the training phase. With the device, a gesture can then be performed in the air. The script reads the incoming set of sensor values and the classifier compares these values to predict the corresponding gesture. The classifier uses its "predict" function to identify the gesture, represented by a number, which is then converted to the corresponding character. The output is stored in a file, which is then read and displayed on the computer screen using Python Tkinter. Figure 13 shows the predicted gesture displayed in the Python Tkinter window.



**Fig.13 Predicted Gesture displayed in the python Tkinter window.**

### Conclusion

The development of an Air Gesture Keyboard using a machine learning approach offers a promising solution for visually impaired individuals to interact with computers more naturally and efficiently. The keyboard uses a depth-sensing camera to capture hand movements, which are then translated into text inputs through advanced machine learning algorithms. By incorporating machine learning and gesture recognition technology, the Air Gesture Keyboard provides visually impaired individuals with an intuitive method of typing through hand gestures. This approach eliminates the need for traditional tactile or auditory-based input methods, offering a more seamless and immersive experience.

The system's technical requirements include an accelerometer (MPU 6050), an Arduino UNO microcontroller, button switches, and a Bluetooth device for wireless connectivity. These components work together to capture and process hand gestures, transmitting the generated patterns to the computer system.

A comprehensive literature survey reveals that hand gesture recognition using machine learning has been explored in various domains. Researchers have utilized different techniques, such as EMG signals and long short-term memory (LSTM) algorithms, to achieve real-time and accurate gesture recognition.

The system design involves creating gesture recognition software using the scikit-learn library in Python. The software employs classification algorithms, such as support vector machines (SVMs), to classify accelerometer data into different gestures. The training set is crucial for accurately recognizing and classifying gestures, and data pre-processing techniques are applied to handle noise and optimize the learning process.

Overall, the Air Gesture Keyboard holds great potential for improving the accessibility and usability of computers for visually impaired individuals. By leveraging machine learning and gesture recognition, this innovative solution can enhance the quality of life for individuals with visual impairments, providing them with a more inclusive and efficient means of communication and interaction. Further research and development in this field can lead to even more advanced and sophisticated gesture recognition systems in the future.

### REFERENCES

- [1]. Clark, J. (2018). Keyboard Layouts and Their Influence on Typing Efficiency. *\*Journal of Computer Science*, 12\*(6), 244-256. DOI: 10.3844/jcssp.2018.244.256
- [2]. Smith, A., & Johnson, B. (2022). Enhancing Text Input with Remote-Based Gesture Keyboards. *\*Proceedings of the International Conference on Human-Computer Interaction\**, 315-321. DOI: 10.1007/978-3-319-95276-3\_32
- [3]. Rodriguez, M., et al. (2021). Development of a Motion Tracking Device for Remote-Based Gesture Keyboards. *\*International Journal of Electrical Engineering and Computer Science*, 5\*(1), 21-30. DOI: 10.15627/jee.2021.0002
- [4]. Wang, Y., & Chen, Z. (2019). Remote-Based Gesture Keyboard: A Novel Approach for Text Input. *\*Journal of Human-Computer Interaction*, 35\*(2), 127-140. DOI: 10.1080/07370024.2018.1566789
- [5]. Shakunthaladevi, M., & Revathi, R. B. (n.d.). Real-time hand gesture recognition using AVR Microcontroller. *\*SKP Engineering College\**, Tiruvannamalai, India.
- [6]. Jaramillo, G., & Benalcázar, M. E. (2017). Real-time hand gesture recognition with EMG using machine learning. *\*Proceedings of the IEEE ETCM\**, 1-5. DOI: 10.1109/ETCM.2017.8247487

ORIGINALITY REPORT

---

7%

SIMILARITY INDEX

3%

INTERNET SOURCES

4%

PUBLICATIONS

2%

STUDENT PAPERS

---

PRIMARY SOURCES

---

1

Shruthi K S, Naveen B, Shivashankara S.  
"Investigating the Impact of Dataset Size on  
Model Complexity for Leaf Disease  
Recognition using CNN", 2023 International  
Conference on Recent Advances in Science  
and Engineering Technology (ICRASET), 2023  
Publication

1%

2

[scholar.lib.ntnu.edu.tw](https://scholar.lib.ntnu.edu.tw)  
Internet Source

1%

3

B C Kavitha, K B Naveen. "Image Acquisition  
and Pre-processing for Detection of Lung  
Cancer using Neural Network", 2022 Fourth  
International Conference on Emerging  
Research in Electronics, Computer Science  
and Technology (ICERECT), 2022  
Publication

1%

4

Submitted to CSU, San Jose State University  
Student Paper

1%

5

Submitted to An-Najah National University  
Student Paper

1%

---



6	Submitted to NIIT University Student Paper	1 %
7	Submitted to Indiana University Student Paper	<1 %
8	<a href="http://www.coursehero.com">www.coursehero.com</a> Internet Source	<1 %
9	Submitted to Purdue University Student Paper	<1 %
10	Jiaqing Zhang, Jie Lei, Weiying Xie, Daixun Li. "Invariant Attribute-Driven Binary Bi-Branch Classification of Hyperspectral and LiDAR Images", Remote Sensing, 2023 Publication	<1 %
11	Shakunthaladevi, M., and R. B. Revathi. "Real time hand gesture recognition using AVR microcontroller", International Conference on Information Communication and Embedded Systems (ICICES2014), 2014. Publication	<1 %
12	<a href="http://waseda.repo.nii.ac.jp">waseda.repo.nii.ac.jp</a> Internet Source	<1 %
13	Dario Ortega Anderes, Ahmad Lotfi, Amir Pourabdollah. "Eating and drinking gesture spotting and recognition using a novel adaptive segmentation technique and a	<1 %

# gesture discrepancy measure", Expert Systems with Applications, 2020

Publication

---

---

Exclude quotes      On

Exclude matches      Off

Exclude bibliography      On

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6