# SMART FLOW CHART EVALUATOR

A.G.Malanga Mishad Vishwajith Thilakarathna
Sri Lanka Institute of InformationTechnology
Colombo, Sri Lanka
it16035836@my.sliit.lk

Mrs. M.P.A.W. Gamage
Departement of Information Technology
Sri Lanka Institute of InformationTechnology
Colombo, Sri Lanka
anjalie.g@sliit.lk

***Abstract*— Abstract— A flowchart is a diagram that is commonly used by people over the world. The purpose of a flowchart is to describe a process with the use of world recognized shapes. There are six main symbols for the important components of the process, the flow of the process is represented by the use of arrow. Due to the mentioned reasons, a flowchart can explain the process even to people who have no knowledge about a process. Because of these factors, flowcharts are widely used in the Information Technology industry by programmers and Information Technology students. This research is done with the aim of reading a flowchart to extract the details and giving a score to evaluate the correctness of the process for a given question. The handwritten flowchart will be compared to a marking scheme and the marks will be generated for the handwritten flowchart accordingly.**

***Keywords— SVM, pseudocode, flowchart***

## I. INTRODUCTION

A flowchart is a diagram which shows a process depicted using universally accepted shapes and text. These diagrams are often simple and small, but they are commonly used since they are easy to understand. It is one of the basic forms of showing a process, also known as one of the seven basic tools of quality control, along with histograms, check sheets, etc. therefore it is taught from school upwards past the university level. It is commonly used in the Information Technology Industry because of its convenience and furthermore used because of its ability to navigate through multiple possibilities and find an output that suits that possibility.

For a flowchart to show a process, it needs different shapes showing different functions. There are four main types of shapes used in a flowchart.

• An activity shape, showing a process, denoted by a rectangle.

• A decision shape, denoted by a diamond.

• A input/output shape, denoted by a parallelogram.

• A start/end shape denoted by a oval.

Other shapes are can be described as in Fig.1.

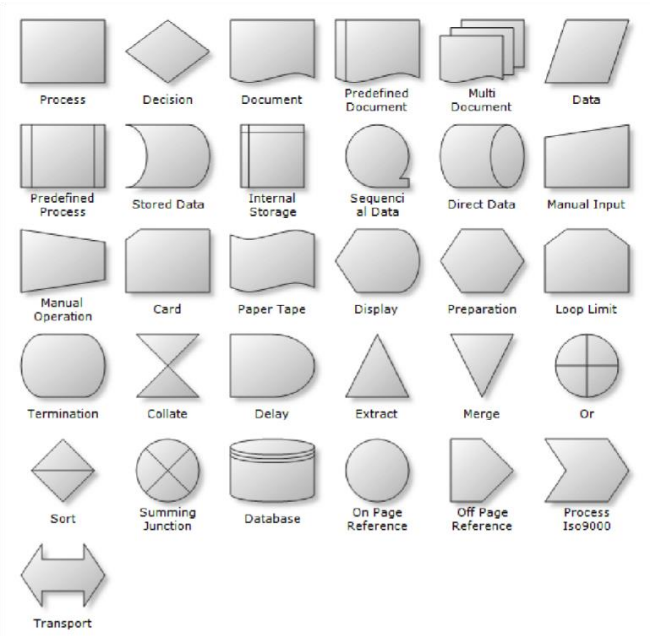


Fig. 1. Flowchart symbols

These Flowcharts are widely used from grade school upwards up to industry level to depict the pathway of a certain process as well as its possible outcomes. For a certain problem, there can be multiple ways to depict the process with the use of a flowchart, and they can all be equally correct. The current research that is happening about the digitalization of flowcharts proposes ways to upload a photo or use an android based app, but none so far uses react-native to make the product available to both android and IOS users. Furthermore, they detect the components and the shapes of the flowchart, but barely recognizes the text and doesn't convert it into a pseudo-code form. In many cases where different flowcharts can depict the same solution, both the flowchart and the pseudo-code should be checked if to know whether it satisfies the solution or not. In a grading system, for an issued problem, there be many different flowcharts that provide the required output to the problem, therefore there can be different marking schemes for the different flowcharts. For a professor that has to grade hundreds of papers, checking across all the different marking schemes for each flowchart can be time-consuming. Furthermore long tiring hours are prone to human errors. There is no system developed for a student to evaluate up to which extent his/her flowchart is incorrect.

## II. OBJECTIVES

To build an offline system in which the user can upload handwritten flowcharts. This eliminates the need for accessories and other requirements such as graphic tablets that such applications often require in order to draw the flowchart into the system. In being offline, it is convenient for the user. The system is to identify text and shapes on its own without the need of human interaction. This also includes loosely drawn diagrams with shifts in the drawing styles due to the unique writing of each user. The system should be able to accurately analyze the uploaded flowcharts without human interaction. Each and every flowchart uploaded should be analyzed and converted to its respective pseudocode form. The system should be able to compare every built pseudocode against marking scheme accurately.

## III. RELATED WORK

FLOW CHART EVALUATOR are still at performance level. Currently, researches have been conducted to simulate the problem as follows,

### *A. Detecting hand-drawn flowcharts using TensorFlow Object Detection API [1]*

Detects the hand-drawn flowchart shapes using TensorFlow Objection Detection.[2]

### *B. On-Line Handwritten flowchart Recognition, Beautification and Editing System.[3]*

This is a beautification and editing method for handwritten and inputted flowcharts using graphic tablets or whiteboards, recognizing the shapes. It proposes a faster method to Microsoft Visio where dragging a symbol through a longdistance is unnecessary, instead of identifying and beautifying the hand-drawn shape. The features of this product are

• Recognize the difference between a graphic shape and a flow arrow line

• Check every closed loop, and if a closed-loop is present distinguish it as a graphic shape • Recognize both the graphic shape and the text.

### *C. A Novel Pen-Based Flowchart Recognition System for Programming Teaching[4].*

This is a research done on evolving technology and the use of a tablet or an electronic whiteboard to input handwritten flowcharts into the system. The features are include as follows,

• Recognition of sketches using SVM-HMM algorithm [5].

• Differentiate between graphic symbols and flow lines.

• Requires the graphic symbol to be drawn in one line.

• Converts the recognized sketches into a C program language.

### *D. Flowchart Plagiarism Detection System: An Image Processing Approach [6].*

This project compares the shape, text, and orientation between two flowcharts and creates a graph in order to identify plagiarism. This tool includes,

• Identifies the basic shapes using an algorithm.

• Creates a graph for comparison.

• Checks for the similarity in the graphs and measures plagiarism.

### *E. Handwritten Flowchart Generator [7]*

This research is focused on digitalizing hand-drawn flowcharts for the convenience of the user. This will be implemented onto Android devices and has the features as follows,

• Edge detection detects the boundaries of the shapes as well as the arrows and separates the components.

• Uses Graphviz [8] to render and recompose the chart.

• Compares between a reference flowchart and evaluates the outcome.

Based on the studies of the existing systems, we found certain research projects that had been carried out that identifies with a similar purpose. Most of the said project does not allow the user to use a picture of the flowchart but requires the user to input the flowchart using a graphic pen-based mechanism directly into the system. [4] As this would be both inconvenient as well as doesn't acknowledge the necessity ownership of a graphic based input method in order to use the application. Our project allows the user to input a jpeg format of a flowchart, whether it's graphically input or handwritten and photographed in order to increase the convenience. This would also allow the user to use their smartphone camera to input the flowchart since this technology will be react-native based.

Many of the aforementioned projects run on the purpose of recognizing text and shape individually to digitize and beautify them only in order to create a neater flowchart that is easier to read and present. This also requires the boundaries of the shapes to be a closed-loop which can be inconvenient depending on the user's handwriting. These use TensorFlow Objection Detection to distinguish between a graphic and text and uses Graphviz to recompose the flowchart. It neither outputs an answer nor does it crosscheck against a mark scheme.

A flowchart plagiarism system was created which detects the similarity of two flowcharts by using directed graphs. This system only detects the similarity based on the used shapes and arrow flow of the flowchart. Our system breaks the flowchart into a pseudocode format where each step can be crosschecked against every possible correct form of the flowchart and marks each step.

A similar novel pen-based teaching system research project has been carried out which uses a hybrid SVH-HMM algorithm [5] for sketch recognition which finally converts it into C program and gives the output. But in this system, it requires the users to directly draw the flowchart into the system using a graphic pen-based whiteboard. Furthermore, the user has to be careful as to draw the shapes in one line into

a closed-loop for the system to detect it. This was created for a teacher to sketch the flowchart into the system. Our project allows the student to input a photographed version of the flowchart which eliminates the need for graphic pen-based whiteboards Furthermore, with the use of Canny technology, our system can detect the shapes using their area. After the student inputs the flowchart it notifies the instructor, and after converting it into a pseudo-code it crosschecks against the marking schemes stored in the repository by the instructor. Furthermore, it cross-checks against all the possible steps from the instructor's code and issues allocated marks for the steps in the student's flowchart respectively.

## IV. METHODOLOGY

The flowcharts are a very common and widely used method of depicting the flow of a process. They are used in different complexities for different purposes, and this project focuses on developing a convenient method of processing these flowcharts. With the increase of the use of smartphone technology, the creation of an application that is available for download on stores for IOS and android phones, the users can quite conveniently use this app as it is both portable and easily accessible.

We introduce two deep learning models to get the best output and increase the accuracy of the flowchart. Once a user uses this application, the flowchart will be converted to pseudocode and cross-checked against a marking scheme, allowing the user to know and further develop their flowchart.

As in Fig. 2. the flowchart and the correct answer id inputted to the Flowchart Evaluator. then the react-native application will be generated the pseudocode for both correct flowchart and the student's flowchart. then both pseudogenes are compared using deep learning models. This flowchart detection system can be broken into 4 main components as follows,

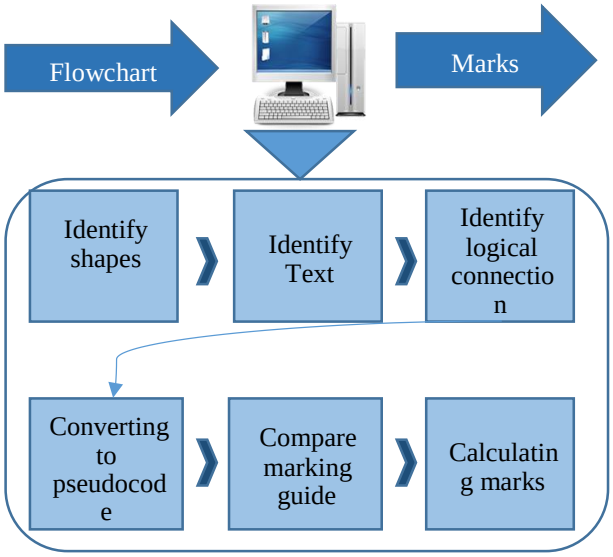


Fig. 2. The architecture of the product

- Pre-processing.
- Shape detection.
- Graph creation.
- Pseudocode comparison.

### A. *Preprocessing of the Flowchart*

The purpose of this component is to detect and identify the shapes of the input flowchart. The flowchart that is input by the user will go through a certain technology that detects the edges of the said flowchart to determine the shapes used present in the input. In order to do so, the edges will be detected by Canny technology [9] which first requires the input image to be converted to a binary image in order to continue with the algorithm. This then identifies the edges of the boundary.

Canny technology is one of the most accurate and popular methods that can be used for the purpose of detecting the edges of the flowchart. steps of the canny detection algorithm

- Removal of the noise of the image by the use of the Gaussian filter.[10]
- Find the intensity gradient of the image
- Use an edge thinning technique in order to eliminate any false or unintended responses to edge detection
- Determine potential edges using double threshold
- Taking into account only the strong edges or edges connected to strong edges
- The arrows from the input flowchart will be removed in order to obtain a more accurate result in shape detection. This process is done in order to isolate the shapes in the flowchart for the shape detection stage.

### B. *Shape Detection*

This component identifies where the basic shapes, such as the eclipse, circle, square and diamond, of the flowchart is used. The shape is determined with the use of an area detection technique. We used a dataset consisting of 1000 individual flowcharts. The first 200 datasets were drawn using tools in order to create perfect shapes and text to increase the detection accuracy of the model. The other 800 were created using 12 different people to create 800 unique flowcharts; in doing so the model will be able to detect different variations of handwriting and shapes. We use mask R-Cnn and inception method for the model. Inception is a new method used to increase the speed and accuracy of a deep learning model. Inception method is computationally cheaper with a higher performance. The architecture of layers are different in comparison to the normal deep learning models where layers are stacked densely in a vertical manner, layers in inception are stacked sparsely in a horizontal manner.

### C. *Graph Creation*

This component creates a directed graph from the identified shapes of the flowchart. The initial stage involves identifying the flow of the process by analyzing how the arrows are connected to each shape in order to create a directed graph. The flow of the graph will be represented by the directed edges while the vertices represent the identified shapes, and the vertex will carry the information regarding the shape and its content. Each vertex is connected by a directed edge, which in turn contains the start and the end of a vertex.

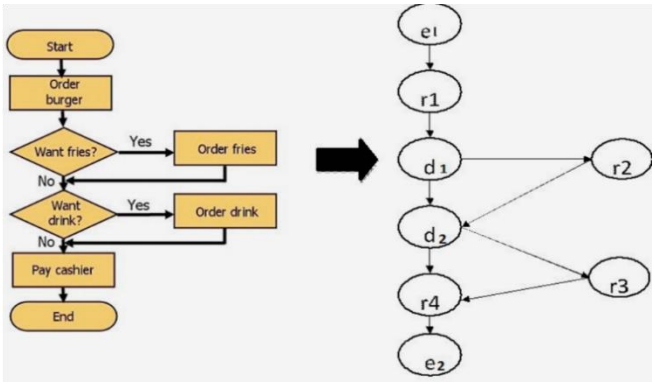


Fig. 3. Graph creation.

### D. *Pseudo Code Comparison*

In this section, the flowchart is converted to a pseudo-code form. As a flowchart can contain different forms that can satisfy the same question, converting the flowchart into a pseudo-code allows the possibility of cross-checking the input against all the possible answers accurately. Initially, the uploaded flowchart/s in which all the input flowcharts will be checked against will be stored in a repository. When the user inputs the flowchart, the marking scheme stored in the repository will be used for correction. After the graph is generated, the system will recognize the graph and generate its respective pseudo code. The pseudo-code of the marking scheme, as well as the input, will be compared, issuing marks assigned for each step.

E.g.: the instructor of a classroom can issue a question for the students which may contain multiple satisfactory answers. The instructor will have to upload the possible answers as a marking scheme which will be stored in the database. Once a student inputs their answer into the system, it will cross-check against the answers in the database issuing the marks as well as notifying the instructor of the students and their answers.

As shown in Fig.3. each component of the Flowchart Evaluator will be connected in a flow to achieve the expected results.

The steps of the pseudocode comparison procedure are as follows.

After shape detection, we crop out the individual shapes from the image and assign an ID to each, and then we detect the text available in the flowchart.

ID Shape

1 → Rectangle

2 → Oval

Once model detects the shapes and assigns the IDs to each shape, it then detects its respective text in ascending order of the IDs, hence detecting text in the flow of the flowchart. The equations of the flowchart will be detected as text, and this will be detected as an equation through the use of the attention model. Attention is a mechanism that was developed to improve the performance of the encoder decoder run on the machine translation. It will convert what was detected as just characters into meaningful mathematical equations.

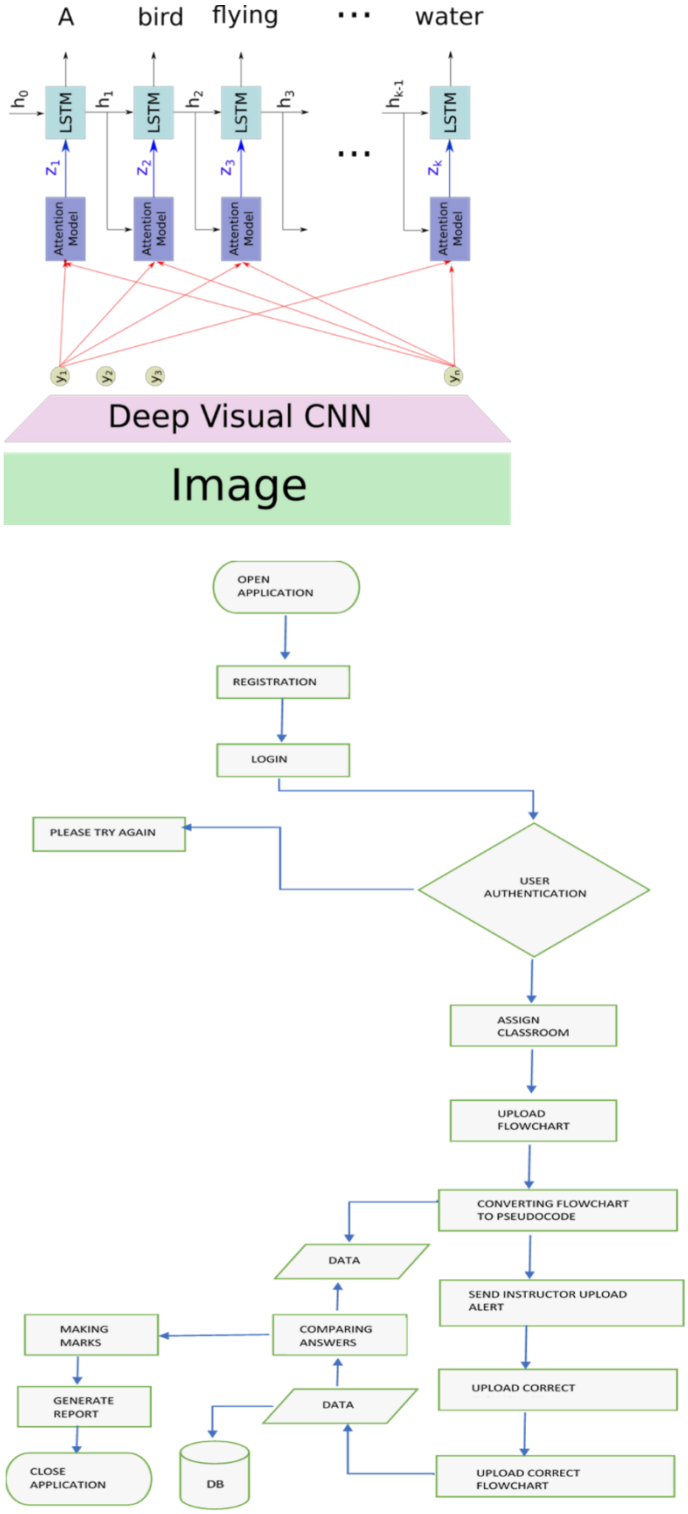


Fig. 4. Flowchart Evaluator Application flow

## V. RESULTS

We have successfully identified and authenticated flowcharts in the aforementioned system. The flowchart evaluator model was successfully calculating the accuracy rate as 97% accurate recognition as shown in the figure below. further specifies show that text analysis accuracy reach up to 96% and shapes accuracy reach up to 98%. Fig.5.

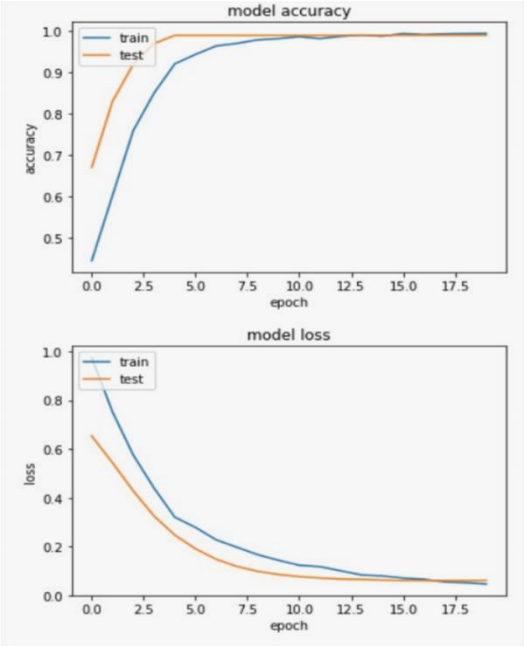


Fig. 5. Flowchart detection accuracy

## VI. CONCLUSION

Flowcharts are known and used for different purposes in different industries all over the world. They are taught grade school upwards to university level because flowcharts are easy to understand and easy to use, depicting the question, process and different solutions available to different approaches. Though flowcharts are taught throughout the learning years of a student, even with advancing technology there is currently no developed method that can conveniently evaluate and correct flowcharts against a marking scheme. In introducing this react native app, we are achieving many objectives which makes the learning process more convenient for students and teachers. By identifying texts and shapes separately, create a pseudocode, train and optimizing the model, we have achieved a system which can accurately identify handwritten flowcharts through image processing, and cross check it against a marking scheme. The flowchart evaluator is a time saving convenient application for both students and the teacher. It takes less effort but accelerates the learning process. With the use of this application, we can further take steps towards virtual learning in parallel to the upcoming technologies of the world.

## REFERENCES

[1] "Ruturaj123/Flowchart-Detection: Detecting hand drawn flowcharts using Tensorflow Object Detection API - Faster RCNN."[Online].Available: https://github.com/Ruturaj123/Flowchart-Detection. [Accessed: 06-Mar-2020].

[2]"Object detection | TensorFlow Lite." [Online]. Available: https://www.tensorflow.org/lite/models/object_detecti on/overview. [Accessed: 06-Mar-2020].

[3] H. Miyao and R. Maruyama, "On-line handwritten flowchart recognition, beautification, and editing system," in Proceedings - International Workshop on Frontiers in Handwriting Recognition, IWFHR, 2012, doi: 10.1109/ICFHR.2012.250.

[4] Z. Yuan, H. Pan, and L. Zhang, "A novel pen-based flowchart recognition system for programming teaching," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2008, doi: 10.1007/978-3-540-89962-4_6.

[5] "Hidden Markov model - Wikipedia." [Online]. Available: https://en.m.wikipedia.org/wiki/Hidden_Markov_mod el. [Accessed: 06-Mar-2020].

[6] J. S. Kuruvila, M. Lal, R. Roy, T. Baby, S. Jamal, and K. K. Sherly, "Flowchart Plagiarism Detection System: An Image Processing Approach," in Procedia Computer Science, 2017, doi: 10.1016/j.procs.2017.09.111.

[7] Z. L. Hsu, Tien-ning, Qian Yu, Rao Zhang, "Handwritten Flowchart Generator," pp. 3–4.

[8] "Graphviz - Graph Visualization Software." [Online]. Available: https://www.graphviz.org/. [Accessed: 06Mar-2020].

[9] R. Deriche, "Using Canny's criteria to derive a recursively implemented optimal edge detector," International Journal of Computer Vision, 1987, doi: 10.1007/BF00123164.

[10] "Gaussian filter - Wikipedia." [Online]. Available: https://en.wikipedia.org/wiki/Gaussian_filter. [Accessed: 06-Mar-2020].