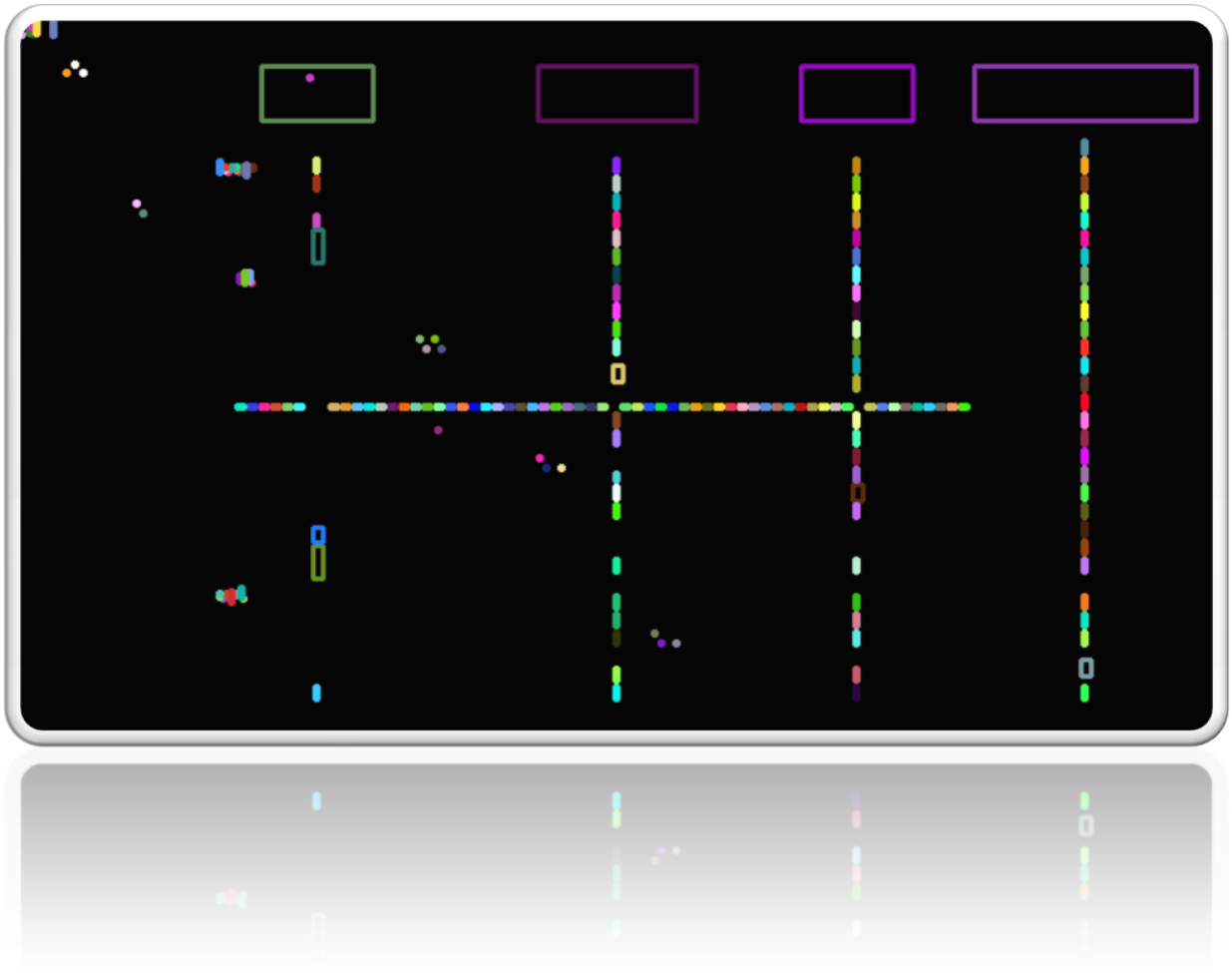
CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG



# Automatic Classification of UML Sequence Diagrams from Images

*Bachelor of Science Thesis in Software Engineering and Management*

Sayf Rashid

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2019

CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG



# AUTOMATIC CLASSIFICATION OF UML SEQUENCE DIAGRAMS FROM IMAGES

**SAYF RASHID**



Examiner: Jan-Philipp Steghöfer

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Cover:
The picture illustrates a sequence diagram image after the findcontours method (a method of the openCV library) has been executed. The method was essential to perform the feature extraction.

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2019

# Automatic Classification of UML Sequence Diagrams from Images

Sayf Rashid
Department of Computer Science and Engineering
University of Gothenburg
Gothenburg, Sweden
gusrashisa@student.gu.se

***Abstract*— Academia's lack of UML artifacts has been an impediment in researching UML and its implication in software development. This has initiated the conception of the UML repository, which is a platform were researchers can share and study UML artifacts. To build such a repository it's required to collect UML diagrams. Therefore, an artifact that can automatically classify such diagrams would be of great value. In 2014, two students of University of Gothenburg successfully developed such an artifact. However, it was limited for classification of class diagrams only. This paper presents an extension of that work by including sequence diagrams, and considering that the most accurate machine learning model in the study was support vector machines, it was decided that further emphasis has to be put on researching support vector machines to maximize its usage to further improve the classifying accuracy. The data elements (feature variables) inputted to the classifier were acquired from the extracted features using image processing. The research was carried out by using a design science approach, which is an iterative research methodology that dictates an evaluation at the end of the iteration.**

***Keywords—UML; machine learning; feature selection; feature extraction; sequence diagrams; image classification.***

## I. INTRODUCTION

### A. The UML Repository

Unified modeling language (UML) is de-facto industry standard approach to illustrate the architecture and the design of the system [1]. Sequence diagram which is a type of UML diagram is predominantly employed to visualize the dynamic behavior of the system and therefore considered an important artifact to understand the system and the software architecture as a whole [2].

This paper is extending previous work on UML classification for class diagrams [3] by including UML sequence diagrams.

In addition to presenting the design of the sequence diagram classifier the paper is a part of a bigger ambition to develop the UML repository, which is a platform to search/retrieve/filter diagrams from a collection of UML images from open source projects [4]. This repository exists to aid researchers in studying and sharing modeling artifacts.

The background of the ambition to develop the UML repository stems from several authors emphasizing the need to study UML practice in software development, the recurring motivation for this need is to better understand how modeling affects the software quality [5].

### B. Research problem

The objective with this paper is to automate classification of images containing sequence diagrams from other images.

The authors of the previous work on class diagrams has already developed an artifact that is capable of extracting most elements of a sequence diagram necessary to build the features needed to be used as feature variables, meaning elements such as rectangles and lines. Furthermore, the same features used as feature variables to classify class diagrams could also be used in this research.

What the main problem is and what this research intend to find out, is if those features are enough, or is it necessary to create new features to accurately classify sequence diagram images from other images.

To answer that an iterative research approach were employed, design science (section III). Were the main goal in the first iteration of the process was to build a dataset (section IV) and train the previous artifact with the new dataset.

### C. Contribution

The papers contribution is:

- The performance of the features extracted for predicting classification of UML sequence diagrams.
- The technical contributions involving realizing the artifact.
- The classifier will aid academia for further research on UML.
- The dataset used in this research could be used to perform further research on sequence diagrams.

### D. Structure of the paper

The remainder of the paper is organized as follows. Section II reviews previous works related to the UML repository and image classification also an in-depth review of the support vector machine (SVM) machine learning model.

Section III presents the research question and the methodology.

Section IV describes the approach employed in this research. Section V, presents the results. Section VI discusses

the results of the research and the last section VIII ends with a conclusion.

## II. Related work & Theory

Despite the lack of research on classification of UML sequence diagrams. There exists a considerable amount of research on image classification in related areas, such as classification of charts and class diagrams [3].

In a similar research conducted on recognizing UML diagrams a tool was developed alongside that takes an image as input and the classification verdict as the output [6].

Though a difference to note is that the tool does not classify the type of UML diagram, it merely recognizes if it is a UML diagram or not.

### A. The UML Repository

The goal of building the UML repository has made a lot of progress, currently img2UML tool has been built that can extract class diagrams images to XMI files [7]. Also a class diagram classifier artifact has been developed [3].

### B. Previous work

The research and work behind the class diagram classifier has been presented in the paper "Automatic Classification of Class Diagrams from Images", which is where most of the ideas and approaches used in this research were derived from.

The authors' goals was to find which features in an image could best classify class diagrams from images and also which machine learning model performs the best.

Their approach was to first identify and then extract the features they deemed hold value in classification.

Once the 23 identified features were retrieved from the image processing (Table I), they developed the classifier by training a machine learner using a labeled training set containing those 23 features as the feature variables (also called predictors).

For the dataset the authors' retrieved 1300 images from Google, 650 of those images were class diagrams, 650 images were non-class diagrams e.g. sequence diagrams and other diagrams, also a few random images were inserted.

Information gain was used to evaluate the performance of the features, were it was evident that F09 carried most information, though it has to be noted that information gain doesn't tell which feature carries most weight when it come to the classification.

The authors also ran the feature sets with different features from the table, were the best performance was achieved when all of the features were used.

Of the six machine-learning models that were evaluated logistic regression (achieved 91.4%) which was the most accurate in classifying non-class diagram images and SVM was the least accurate (achieved 89.0%). Since the priority was to exclude non-class diagrams, thus the best models were the one with highest rate in classifying non-class diagrams.

Therefore, logistic regression was considered to be the best, respectively SVM the worst machine learning model for classification of class diagrams from images. Though the same authors in their bachelor thesis [8] found out in the same research that SVM performs the best in classifying non-class diagrams, were it achieved 92.90%.

Table I. Extracted features

| Feat. | Name | Description |
|---|---|---|
| **F01** | *Rectangles' portion of image, percentage* | Calculated by dividing the sum of the area of all the rectangles with the area of the image itself |
| **F02** | *Rectangle size variation, ratio* | Calculated by dividing the rectangle size standard deviation with the rectangle average size |
| **F03-06** | *Rectangle distribution, percentage* | The image is divided into four equally sized sections and the area of the rectangles inside the sections is then divided by the total area of the rectangles. The 4 sections sum up to 100% |
| **F07** | *Rectangle connections, percentage* | Calculated by counting all rectangles that are connected to at least one rectangle, and dividing that number by the total amount of rectangles in the image |
| **F08-10** | *Rectangle dividing lines, percentage* | The rectangles are split into three groups, with rectangles that have: no dividing lines (F08); one or two dividing lines (F09); or three or more dividing lines (F10). This produces three numbers that represent the percentage of rectangles within each group |
| **F11/F12** | *Rectangles horizontally/vertically aligned, ratio* | Sides of rectangles, horizontal (F11) and vertical (F12), that are aligned with sides of other rectangles are counted. The numbers are then divided with the number of detected rectangles in the image -- resulting in two ratios on rectangle horizontal and vertical alignments |
| **F13/F14** | *Average horizontal/vertical line size, ratio* | Average size of horizontal (F13) and vertical (F14) lines that are larger than ⅔ of the images width or height, divided by the images width or height, respectively |
| **F15** | *Parent rectangles in parent rectangles, percentage* | Rectangles that have rectangles within them can possibly be packages. This feature is the percentage of the area of those parent rectangles that is within other parent rectangles |
| **F16** | *Rectangles in rectangles, percentage* | This feature is calculated in the same manner as F15, but with rectangles, instead of parent rectangles |
| **F17** | *Rectangles height-width ratio* | The average ratio between the height of the rectangles and the width of the rectangles |
| **F18** | *Geometrical shapes' portion of image* | The same as F01, but with rhombuses, triangles and ellipses |
| **F19** | *Lines connecting Geometrical shapes, ratio* | The number of connecting lines from shapes, other than rectangles, divided by the number of detected shapes in the image |
| **F20** | *Noise, percentage* | Detected lines that are outside of rectangles, divided by the number of all detected lines |
| **F21-23** | *Color frequency, percentage* | Three most frequent colors in the image are found. Then a percentage out of all appearing colors is found for the three colors |

*Note. Table retrieved from "Automatic classification of UML Class diagrams from images", by T. Ho-Quang & M. R.V. Chaudron, 2014.*

### C. Image classification

There are various image classification approaches [9, 10, 11]. This subsection will elaborate on approaches and theories relevant for this research.

An image classification process could be followed through the following steps [12]:

*1) Identification of required data*

Obtaining significant amount of data is critical to accurately develop a classifier [11]. According to [11] the optimal approach to collect the required dataset is through an expert. Otherwise brute-force is the best choice, i.e. collect any relevant data, though this approach expects a considerable amount of pre-processing.

*2) Preprocessing*

Often the dataset and the images contain "noise" and "inconsistencies". Therefore, a preprocessing step before the image extraction is highly preferred [10].

The preprocessing steps may involve normalization and filtering [9].

*3) Feature extraction*

An important step in image classification is feature extraction, were the goal is to extract the most important feature that discriminate it from other classes [13]. According to [14] a feature has a large discriminative power if it is similar within the same class but dissimilar between different classes.

*4) Machine learning model selection*

Choosing the correct machine learning model is a difficult task, though support vector machine has given great result in classification of class diagrams [8] and has a good reputation in achieving high accuracy with a small training-set [15], which makes it the best candidate model for this research.

*5) Evaluation of classification performance*

There are different aspects to look for when evaluating a classifier the most common being the accuracy [12].

### D. Machine learning

Machine learning which belongs to the domain of AI has due to its increasing popularity recently made a lot of progress. This popularity can be credited to the surge in data usage in all areas of the society [16]. One area of particular interest to this research is image classification.

Although there are different types of machine learning approaches, such as reinforcement learning and unsupervised learning, this research will only focus on the supervised learning approach.

In supervised learning, the machine learner learns by the manually labeled data, which is also called the training set. I.e., when training the machine-learner every data is mapped to its corresponding label. Through this procedure the machine-learner learns a behavior or a pattern [17].

There are various machine-learning models that suit its specific purpose [16].

### E. Support Vector Machine

*1) Background*

Support vector machine (SVM) is a supervised machine learning model, which has its roots from statistical learning theory and structural risk minimization principle [15]. Its popularity in classification problems is mainly accredited to its simplicity and strong performance [15, 16, 18].

SVM operates by finding the optimal hyper-plane that separates the classes.

The hyper-plane is given by this formula:

$$\boldsymbol{f(x) = w^T \times x + b}\,.$$

Were (w) is the weight vector, (x) the input vector and (b) the bias.

To find the optimal hyper-plane that separates the classes it is needed to look (mathematically) were the vectors (input data from the training-set) are closest to the hyper-plane f(x) from the two classes that have the biggest margin also called the maximum-margin line. Those vectors are called the support vectors which will be used to classify the unlabeled data, hence the name support vector machine.

To best explain how SVM operates an example is illustrated in **Figure 1**. The example depicts a two dimensional feature plane (Vertical lines percentage & Horizontal line percentage). We can observe two classes the red class (the data we want to classify), which is labeled with 1 and the blue class which is labeled with -1. The green line is the optimal line that perfectly separates the two classes. The space between the black lines is the maximum margins between the classes. The data on the left side of the margins are classified negative and the red on the right side are classified positive.

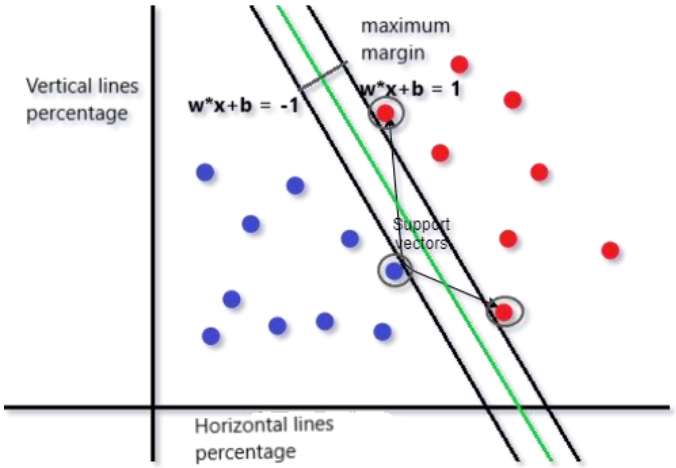



**Figure 1:** A depiction of how a linear SVM separates the data in the feature plane.

The purpose of wanting a vast margin between the classes is to reduce generalization errors and thus increase the accuracy of the classifier [19].

*2) Radial basis function kernel*

Figure 1 depicts a linear classifier which performs well in a linear separable dataset but on non-linearly separable datasets the performance will be unsatisfactory.

Therefore, kernels that map the data into another space were introduced to solve the non-linear problem, such as the radial basis function (RBF) kernel.

If uncertain of which kernel to use the preferred kernel is RBF. Since the kernel is known to perform well in on a variety

of problems [20], except for when the feature space is vast, e.g. text classification. Then a linear kernel is more suiting [21].

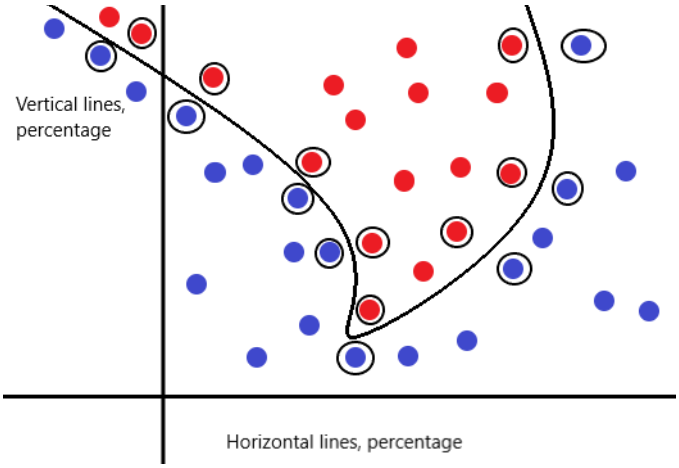



**Figure 2:** A depiction of how RBF kernel separates the non-linear data.

The RBF kernel is given by this formula:

$$K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2 + c}$$

*3) SVM Parameters*

In the RBF formula, the parameter C is the cost which decides how much errors are permitted. The parameter γ is gamma which decides how fit the decision boundaries should be. A large gamma value will lead to a narrow decision boundary**.** Cross-validation is used to tune the parameters, since it is not known beforehand which parameter value performs the best [22].

## III. RESEARCH METHOD

The aim of the presented study is to aid academia and researchers in studying UML artifacts by developing a sequence diagram classifier and thus enriching the UML repository with sequence diagrams.

The aim is broken down into the following research questions:

**Main Research Question:**

**RQ1:** How can classification of UML sequence diagram from images become automated?

**Sub-Questions:**

**SQ1:** What features in an image can help classify an UML sequence diagram, or exclude similar images?

**SQ2:** What level of accuracy can be expected with said classification?

The main research question is addressed by the development of the artifact. The sub questions will address the details to achieving an acceptable accuracy level.

### *Design science research methodology*

Considering the research question and the artifact that needed to be developed and subsequently evaluated in an iterative manner, design science research methodology deemed to be a suitable fit for the research methodology of this thesis.

Design science has the goal of developing an artifact that addresses a practical problem that hasn't been solved before [23]. The remaining part of this section will describe the design science process that was followed:

### *A. Problem Identification*

The first step in our research was to identify the research problem and justify the value of solving the problem.

In section I it was concluded that researchers has a need to study UML artifacts which could be solved through the development of the UML classifier by expanding the UML repository with additional UML diagrams, i.e. sequence diagram images.

Thus, the identified problem is the lack of UML artifacts and part of the solution is the development of the classifier which could help in collecting UML artifacts.

### *B. Objectives of a solution*

The next step in our process was to define the objectives of the sequence diagram classifier. I.e., what the development of the artifact needs to accomplish and which requirements do the artifact need to fulfill in order to be considered accomplished.

Considering that the classifier is an extension of the previous work to classify class diagrams [3, 8], which were used to classify class diagram images from a database containing a vast amount of images. The same priority of having an accuracy rate over 90% in classifying negative images (also called specificity), and achieving over 85% accuracy rate in positive images remains (also called sensitivity). This priority is because the negative images outnumber the positive images.

### *C. Design & Development*

This step along with the evaluation step (D. Evaluation) was iterated until the objective set in B (Objectives of a solution) had been reached. This step entails the development of the artifact and considering that it was developed in multiple iterations, it will be communicated in section IV according to the iterations that were followed.

### *D. Evaluation*

To evaluate the performance of the classifier after the design and development iteration phase was over, 10-fold cross evaluation were employed.

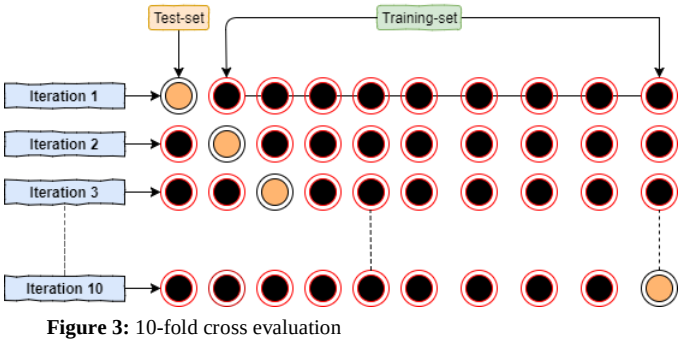



**Figure 3:** 10-fold cross evaluation

K-fold cross evaluation is a statistical method were each image is utilized both in training and for testing the classifier.

The method works by splitting up the dataset into k equally sized folds, and then train on k-1 folds and test the classifier with the last fold. This method is employed to ensure accurate accuracy rate [24].

It has to be noted that k-fold cross evaluation was also used to evaluate and choosing the best parameters for the SVM, though this process was automated by the openCV library method trainAuto.

For evaluating the test-sets, the metrics specificity, sensitivity and accuracy were employed. Specificity also called true negative rate (TNR) were used to evaluate how accurate the classifier classifies non-sequence diagrams. The metric has the following formula:

$$\boldsymbol{Specificity = TNR = \frac{TN}{TN+FP}}$$

Were TN is true negative and FP is false positive, i.e. images that are negative but the classifier classifies it as positive.

Sensitivity also called true positive rate (TPR) were used to evaluate how accurate the classifier classifies sequence diagram images. The metric has the following formula:

$$\boldsymbol{Sensitivity = TPR = \frac{TP}{TP+FN}}$$

Were TP is true positive and FN is false negative, i.e. images that are positive but classified as negative.

Accuracy is the overall correctness of the classifier, i.e. both the false negative and the false positive are measured. The metric is given through the following formula:

$$\boldsymbol{Accuracy = \frac{TP+TN}{TP+TN+FP+FN}}$$

The results of the evaluation are presented in section V (Results).

### *E. Communication*

The research will be communicated through this paper.

## IV. Approach

This section will describe the approach employed in this research, also the features extracted and the motivation for their extractions. Furthermore, an overview of the artifact and the components will be presented.

Before going into details about how iteration 1 were conducted, it has to be emphasized that this research is an extension of previous work [3, 8] and large parts of the previous work were reused with some modification to work for this research.

### *A. UML diagram classifier*

The UML diagram classifier was developed in 2014 by two students of Gothenburg University [8]. It's comprised of two components an image processing, and a machine learning component.

#### *1) Image processing*

The image processing component takes in an image of various formats, which then gets processed with the usage of the popular OpenCV library. In order to extract the features, the coordinates of the lines and corners of the rectangles needs to be located and stored.

This is done by using algorithms such as probabilistic Hough lines transform, canny edge detector and findcontours.

Canny edge detector which is used to detect the edges was used before applying the algorithms to find the lines and contours.

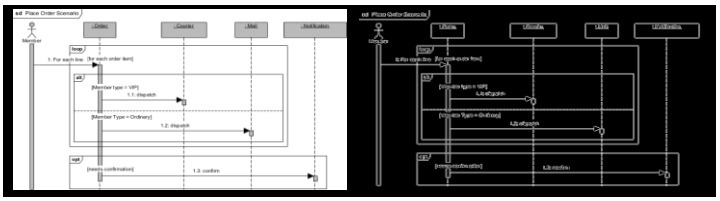


**Figure 4:** Using canny edge detector on a grayscaled image.

Probabilistic Hough lines transform **Figure 5** is used to extract the lines from the images and findcontours **Figure 6** is used to extract the shapes, e.g. the rectangles.

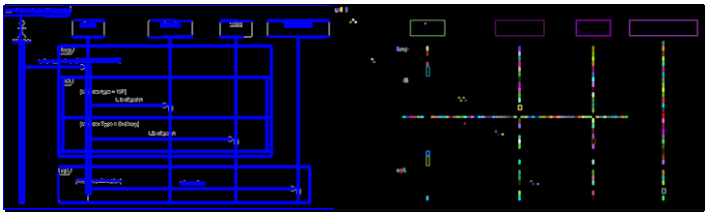


**Figure 5:** Probabilistic Hough lines transform.

**Figure 6:** FindContours method.

Once the image has been processed (**Figure 7)** and the relevant information has been retrieved and the duplicate features has been filtered out, then the building of the feature variables will start in the sub-component feature-extractor.

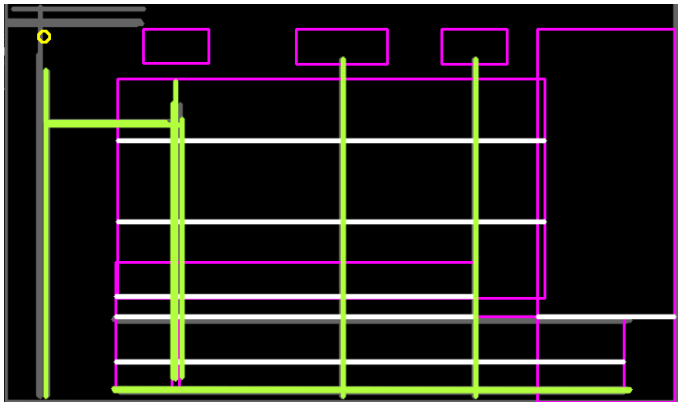


**Figure 7:** The extracted features are put back and displayed with some distorted parts.

#### *2) Machine learning*

The machine learning component is comprised of two parts:

1. The trainer, which takes in a vector with the feature variables gotten from the image processing and maps it into a manually assigned label, meaning a value that represent if it is a sequence diagram or not.

The output of this component is the trained classifier which will be used to classify the unlabeled data.

2. The trained classifier, which takes in unlabeled data and predict whether the data is a sequence diagram or not.

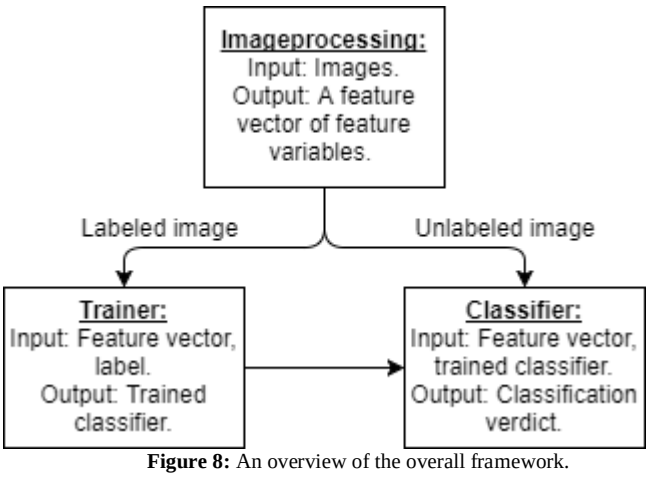



**Figure 8:** An overview of the overall framework.

## B. *Iteration 1*

The plan to conduct the first iteration was largely based on how an image classification process was followed according to section II.

### *1) Identification of required data*

We know from section II that SVMs discriminates between the classes by creating support vectors from the training-set were the features of the respective classes are most similar. Thus, to create the best classifier it is required to have a good representation of sequence diagram images and also of images that have similar features as sequence diagrams.

We chose to obtain a balanced dataset (the negative and positive labeled classes are similarly sized) since SVM performs poorly on imbalanced dataset [25].

The images of sequence diagrams were collected with the usage of the search engine Google by inputting the term "sequence diagram".

We also obtained few sequence diagrams images from the previous works negative dataset for a total accumulation of 375 sequence diagram images. For the negative class we used the previous work negative dataset, since they had a good representation of many different diagram types. Though, the images of sequence diagrams were removed and a portion of the class diagrams from their positive labeled dataset were inputted to our negative labeled dataset.

### *2) Preprocessing*

Before doing the feature extraction the images needs some preprocessing to enhance the features relevant to the feature extraction, e.g. the lines, corner. To perform this procedure the openCV methods grayscale and Gaussian blur were used.

Grayscale is a method that transforms the original image which is usually in RGB (colored) to black & white, it is done because of the methods to extract the features work better in grayscale format.

Gaussian blur is used to blur the image, this is needed e.g. to avoid counting double lines were only one line exist, it is also used to even out the image in case the corners are non-continuous because of low quality.

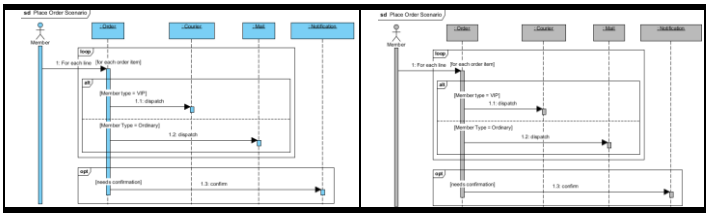


**Figure 9:** From RGB colors to black & white.

### *3) Feature extraction*

In [3, 8] the researcher extracted 23 features (Table I) for classification of class diagram, this paper will explain why the same features are relevant for classification of sequence diagrams.

Firstly we need to be reminded that support vectors are created from both classes training data. Therefore, features that exist in both classes, but more pronounced or less pronounced in one class have more discriminative power.

We also know that SVM uses combinations of the features to discriminate between the classes. Hence, a single feature that seem to hold no value in the classification might play a vital role in the classification performance, when used in combination with other features.

The following features (described in Table I) were used in iteration 1:

**F01:** The size of the image that are rectangles is relevant when used with other features since knowing how much image is covered by rectangles does give some information about the image.

**F02:** The size variation of the rectangles could carry some information for the classifier.

**F03-06:** In which section of the image the rectangles are located in is a relevant feature.

**F07:** Rectangle connection is not prevalent in sequence diagrams but is in other diagrams and therefore relevant.

**F08-10:** F09 was the feature which gave most information in classification of class diagrams, thus, the features is relevant.

**F11/12:** Horizontally aligned rectangles are a defining feature of sequence diagrams and other diagram types.

**F13/14:** The lines that are vertically or horizontally aligned are a relevant feature.

**F15:** Rectangles that are within other rectangles is not prevalent feature in sequence diagrams but is in other diagram types, therefore relevant in classification.

**F16:** Could be relevant.

**F17:** Rectangles height/width ratio a relevant feature.

**F18:** Shapes portion of the image could be a relevant feature.

**F19:** Connecting lines to a shape is a relevant feature since the lines in sequence diagram and other diagram types may have a triangle head.

**F20:** The feature Noise contained information in classifying class diagrams [3] and therefore relevant.

**F21-23:** Probably not relevant, though certain diagrams do have varied color frequency.

The logic behind why the feature variables are either in ratio or in percentage is because the value should be relative to all images. E.g. the amount of rectangles in an image is not relevant, since it does not give enough information. Though, the percentage of the image size that are rectangles is relevant since it gives more information since it tells the machine learner what sort of information the image contains.

*4) Machine learning*

SVM with the RBF kernel performed well in [8] and therefore chosen as the machine learning model. Though there were other SVMs that were considered e.g. linear but since the data type was unknown, RBF was chosen. Primarily because it is the most popular kernel type and performs well in most problem settings. The RBF kernel has two parameters that should be correctly tuned for optimal performance the gamma and the cost as mentioned in section II. The tuning was done automatically by using the openCV method trainAuto, which works in this manner. The method grid searches different parameters iteratively through cross-validation and chooses the parameters that yield the best performance.

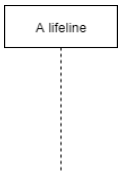
*5)* Evaluation

To evaluate the classifiers performance we followed the evaluation steps described in section III (RESEARCH METHOD).

### *C. Iteration 2*

The requirements were not met in iteration 1, therefore in iteration 2, we needed to define new features that have a large discriminative power and extract them. Though, the previous 23 features were kept, since those features gave good results and were very close to meeting the requirements.

**F24:** By looking at the sequence diagram (Figure 10) we can quickly observe a feature (A lifeline, the picture to the right) that could be used to increase the accuracy of the classifier.




To extract a feature that is more unique to sequence diagrams and thus increase its discriminative power, the feature requires more than one lifeline to be counted as a lifeline, since a sequence diagram contain multiple lifelines. Furthermore the heights of the top rectangles in the lifelines have to be approximately the same.

To increase the distinctiveness of the feature, the lines including the activation rectangle (the vertical rectangles in figure 10) have to be longer than half the height of the image, since the lifeline usually starts from the top of the image and ends at the bottom of the image.

There were certain constraints needed to be considered when selecting this feature:

1. Not all lifelines contain a straight line or dotted line some contain an activation rectangle.
2. The rectangles of a lifeline don't always have to be horizontally aligned (this constraint was ignored since most of the rectangles are horizontally aligned).
3. Not all sequence diagram images uses a rectangle to illustrate the head of the lifeline (this constraint was also ignored since it was considered impossible to include all shape types, and rectangles are the most common).

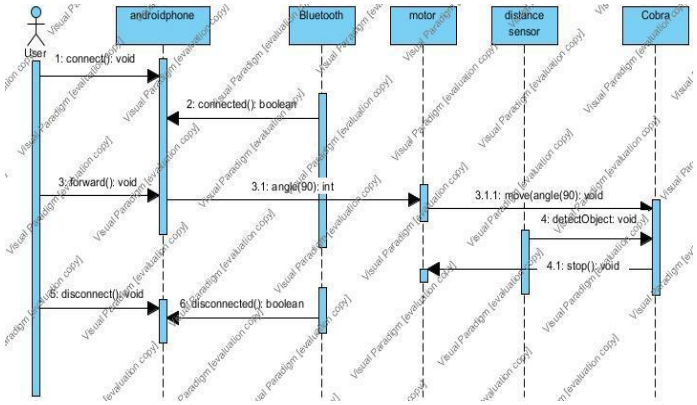



**Figure 10**: A sequence diagram made by the students of SE&M in GU.

**F25:** This feature also targets the same feature as F24 though the line is excluded from the requirement, this feature was chosen by considering that image processing is not perfect and often important details of the diagram are lost, e.g. the lines of the lifelines. To make this feature more unique the bottom corners of the rectangles that are extracted need to be above the center point of the image.

The following table containing the features that were extracted:

TABLE II. FEATURES ADDED

| Feat. | Name | Description |
|---|---|---|
| **F24** | *Lifelines, percentage.* | Percentage of rectangles that are horizontally aligned and have the same height and are connected to a line/dotted line, which is longer than half the image height. |
| **F25** | *Rectangles top of the image that are horizontally aligned, percentage.* | Percentage of rectangles that are aligned horizontally and in the upper part of the image. |

To extract the feature "F24" the first thing that needed to be done was storing all the rectangles, secondly we needed to store all the rectangles containing a line that starts from its southern center point and contains at least 50% of the pixel height of the image, meaning if an image is 500 pixel in height then the line needs to be over 250 pixels.

Counting the height of the line an algorithm originally developed for lane-following in the self-driving car was used though with slight modifications [26]. The algorithm worked like this:

1. Canny edge detector needs to be used for step 2.
2. The lines need to be thickened which is done with probabilistic Hough line.
3. To count the size of the line it's required to have two loops, one that loops the image vertically and another that loops horizontally. The vertical loop begins from where the rectangle ends and continues until the image ends. The horizontal loop is where the algorithm counts the white pixels, which it does by checking the five pixels to the left of the center point of the rectangle, if it detects a white pixel than the algorithm counts it as a pixel belonging to a line.

After confirming if a vertical line exists it was needed to filter out the rectangles that are approximately of the same heights. Thus, the algorithm has established that the rectangle is a lifeline.

In order to make the feature to be a value that is relevant for the machine learner, the percentage of the total rectangles that are the top of the lifelines is computed and inputted to the machine learner.

To extract feature "F25" the steps containing the line was skipped. Furthermore, the rectangles are filtered based on their bottom vertical position, meaning if the y-value is lower than the y-value of the center point it will not be counted as a rectangle. The motivation for removing the rectangles that are beneath the center point of the images is because rectangles in sequence diagrams are predominantly located in the upper part of the image.

## V. Results

This section will present the results based on the iterations that were followed.

TABLE III. CROSS-EVALUATION RESULTS 23 FEATURES

| Iteration | Accuracy | TNR | TPR | FP | FN | Size |
|---|---|---|---|---|---|---|
| **1** | *87.1 %* | *88.1%* | *86.0%* | *5* | *6* | *74* |
| **2** | *88.1%* | *88.1%* | *88.1%* | *5* | *5* | *74* |
| **3** | *93.7%* | *92.5%* | *94.9%* | *3* | *2* | *74* |
| **4** | *92.7%* | *95.0%* | *90.5%* | *2* | *4* | *76* |
| **5** | *89.4%* | *84.4%* | *95.0%* | *7* | *2* | *76* |
| **6** | *89.4%* | *86.4%* | *92.7%* | *6* | *3* | *76* |
| **7** | *84.4%* | *80.9%* | *88.4%* | *9* | *5* | *76* |
| **8** | *90.5%* | *82.6%* | *100%* | *8* | *0* | *76* |
| **9** | *85.4%* | *80.9%* | *90.5%* | *9* | *4* | *76* |
| **10** | *93.5%* | *92.3%* | *94.7%* | *3* | *2* | *72* |
| **TOTAL** | **89.3%** | **86.8%** | **91.9%** | **57** | **33** | **750** |

Table III displays the results gotten from the 10 fold cross-evaluation in iteration 1, were the 23 features from the previous work were evaluated. Size was the amount of images in the evaluations test-set.

TABLE IV. CROSS-EVALUATION RESULTS 25 FEATURES

| Iteration | Accuracy | TNR | TPR | FP | FN | Size |
|---|---|---|---|---|---|---|
| **1** | *93.7 %* | *97.4%* | *90.2%* | *1* | *4* | *74* |
| **2** | *91.4%* | *94.9%* | *88.1%* | *2* | *5* | *74* |
| **3** | *89.2%* | *92.5%* | *86.0%* | *3* | *6* | *74* |
| **4** | *90.5%* | *92.7%* | *88.4%* | *3* | *5* | *76* |
| **5** | *89.4%* | *90.5%* | *88.4%* | *4* | *5* | *76* |
| **6** | *89.4%* | *95.0%* | *84.4%* | *2* | *7* | *76* |
| **7** | *88.4%* | *86.4%* | *90.5%* | *6* | *4* | *76* |
| **8** | *92.7%* | *90.5%* | *95.0%* | *4* | *2* | *76* |
| **9** | *91.6%* | *88.4%* | *95.0%* | *5* | *2* | *76* |
| **10** | *92.3%* | *90.0%* | *94.7%* | *4* | *2* | *72* |
| **TOTAL** | **90.8%** | **91.7%** | **89.9%** | **34** | **42** | **750** |

Table IV displays the results gotten from the 10 fold cross-evaluation in iteration 2 were the 2 features were added to the previous features.

## VI. Discussion

With the results obtained from section V, we can conclude that the requirements of over 90% in specificity and 85% in sensitivity were met. Thus, the research succeeded in achieving automatic classification of sequence diagrams from images.

Due to answer RQ1 and SQ1 our first theory was that the same features to classify class diagram images would also be able to classify sequence diagram images. The results showed that the same features indeed gave good results in achieving classification of sequence diagram from images.

The result of iteration 1 using the 23 features defined in [8] to classify class diagrams gave very similar results as in [3] which too would not meet the requirement of over 90% in specificity.

In iteration 2, two new features were defined based upon the research regarding feature extraction, described in section II.

The goal was to define features that have large discriminative power, which is a feature that is similar within the same class but dissimilar between different classes. Hence, the feature lifeline was extracted. This improved the specificity of the classifier at the expense of the sensitivity.

While examining the two classifiers we can observe a major difference:

TABLE V. SVM PARAMETERS

| | 23 Feats. | 25 Feats. |
|---|---|---|
| **Gamma** | 0.03375 | 0.50625 |
| **Cost** | 6.25 | 6.25 |

The difference in the gamma value, which affects the decision boundary, is substantial. This could be the reason why the 23 features classifier has higher sensitivity and lower specificity. As we know from section II (SVM) that lower gamma results in larger decision boundary. This explains why the big fluctuation in accuracy, 9.3% in iteration 1 versus 5.3% in iteration 2.

Considering that the trainAuto method uses the best parameters as possible we can conclude that the two added features do improve the accuracy.

### A. Threats to validity

#### 1) Threats to Internal Validity

The image processing component is a validity threat since there is no possible way to process the image to suit all images. I.e. different images require different preprocessing algorithms and thresholds to enhance the features. Therefore some images will always have less or wrong features extracted which can affect the classification verdict.

#### 2) Threats to External Validity

The size of the dataset were much smaller in this research (750 images) compared to the previous works dataset (1300 images) which could reduce the classifiaction accuracy and therefore considered a validity threat.

## VII. Conclusion & Future work

With the accuracy of the classifier meeting the requirements, this paper also found that the same features used in classification of class diagrams yielded good result in classifying sequence diagram images.

The contribution of this research is the sequence diagram classifier and the results of the features performance in classification of sequence diagrams from images.

For future work on classification of sequence diagrams the performance of the machine learning model neural networks could be evaluated and compared to SVM.

## Acknowledgment

The author of this paper would like to acknowledge and thank the work and efforts of all who have contributed to the making of this paper.

## References

[1] G. Scaniello, B. Noble, and I. N. Sneddon, "On the Impact of UML Analysis Models on Source Code Comprehensibility and Modifiability,", April 2014 G. Scaniello, B. Noble, and I. N. Sneddon, "On the Impact of UML Analysis Models on Source Code Comprehensibility and Modifiability,", April 2014

[2] R. Rathinasabapathy, "Object oriented software design for association rule mining algorithms using sequence diagram," 2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, 2015, pp. 1-4.

[3] T. Ho-Quang, M. R. V. Chaudron, I. Samúelsson, J. Hjaltason, B. Karasneh, H. Osman, "Automatic classification of UML class diagrams from images", *Proceedings of the 2014 21st Asia-Pacific Software Engineering Conference - Volume 01*, pp. 399-406, 2014.

[4] The UML repository. http://models-db.com/

[5] B. Karasneh, M. R. V. Chaudron, "Img2UML: A System for Extracting UML Models from Images", *39th Euromicro Conference Series on Software Engineering and Advanced Applications Santander*, 2013.

[6] Valentín Moreno, Gonzalo Génova, Manuela Alejandres, and Anabel Fraga. 2016. Automatic classification of web images as UML diagrams. In Proceedings of the 4th Spanish Conference on Information Retrieval (CERI '16). ACM, New York, NY, USA, Article 17, 8 pages.

[7] B. Karasneh and M. R. V. Chaudron, "Extracting UML models from images," *2013 5th International Conference on Computer Science and Information Technology*, Amman, 2013, pp. 169-178.

[8] I. Samúelsson, J. Hjaltason, "Automatic classification of UML Class diagrams through image feature extraction and machine learning" in bachelor thesis, 2015.

[9] Choras, Ryszard S.. (2007). Image feature extraction techniques and their applications for CBIR and biometrics systems. International Journal of Biology and Biomedical Engineering. 1.

[10] Shameena N, R. Jabbar, "A Study of Preprocessing and Segmentation Techniques on Cardiac Medical Images", in International Journal of Engineering Research & Technology (IJERT), 2014.

[11] Kotsiantis, Sotiris. (2007). Supervised Machine Learning: A Review of Classification Techniques.. Informatica (Slovenia). 31. 249-268.

[12] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," International Journal of Remote Sensing, vol. 28, no. 5, pp. 823-870, 2007.

[13] G. Kumar and P. K. Bhatia, "A Detailed Review of Feature Extraction in Image Processing Systems," *2014 Fourth International Conference on Advanced Computing & Communication Technologies*, Rohtak, 2014, pp. 5-12.

[14] A. Vailaya, A. Jain and Hong Jiang Zhang, "On image classification: city vs. landscape," *Proceedings. IEEE Workshop on Content-Based Access of Image and Video Libraries (Cat. No.98EX173)*, Santa Barbara, CA, USA, 1998, pp. 3-8.

[15] Achirul Nanda, Muhammad & Seminar, Kudang & Nandika, Dodi & Maddu, Akhiruddin. (2018). A Comparison Study of Kernel Functions in the Support Vector Machine and Its Application for Termite Detection. Information. 9. 5. 10.3390/info9010005.

[16] Osisanwo F.Y., Akinsola J.E.T., "Supervised Machine Learning Algorithms: Classification and Comparison", in International Journal of Computer Trends and Technology(IJCTT) –Volume 48 Number 3 June 2017.

[17] Hsu, C.-W & Chang, C.-C & Lin, C.-J. (2003). A Practical Guide to Support Vector Classification. 101. 1396-1400.

[18] Simeone, Osvaldo. (2018). A Very Brief Introduction to Machine Learning With Applications to Communication Systems.

[19] Andrew Ng, "Support Vector Machine, CS229 Lecture notes".

[20] Claesen, Marc & De Smet, Frank & Suykens, Johan & De Moor, Bart. (2014). Fast prediction with SVM models containing RBF kernels.

[21] R. Toth, "Selecting Negative Examples for Training an SVM Classifier," in Master thesis, 2011.

[22] Amami, Rimah & Ben Ayed, Dorra & Ellouze, Noureddine. (2015). Practical Selection of SVM Supervised Parameters with Different Feature Representations for Vowel Recognition. 7.

[23] Hevner, Alan & R, Alan & March, Salvatore & T, Salvatore & , Park & Park, Jinsoo & , Ram & , Sudha. (2004). Design Science in Information Systems Research. Management Information Systems Quarterly. 28. 75-.

[24] P. REFAEILZADEH, L. Tang, "Cross-Validation" in Encyclopedia of Database Systems, 532-538, 2009.

[25] Akbani, Rehan & Kwek, Stephen & Japkowicz, Nathalie. (2004). Applying Support Vector Machines to Imbalanced Data Sets. Lecture Notes Artif. Intell.. 3201. 39-50. 10.1007/978-3-540-30115-8_7.

[26] Bacha, Andrew. (2019). Line detection and lane following for an autonomous mobile robot [electronic resource] /.