Software Engineering

Princess Sumaya University for Technology
جامعة الأميرة سمية للتكنولوجيا

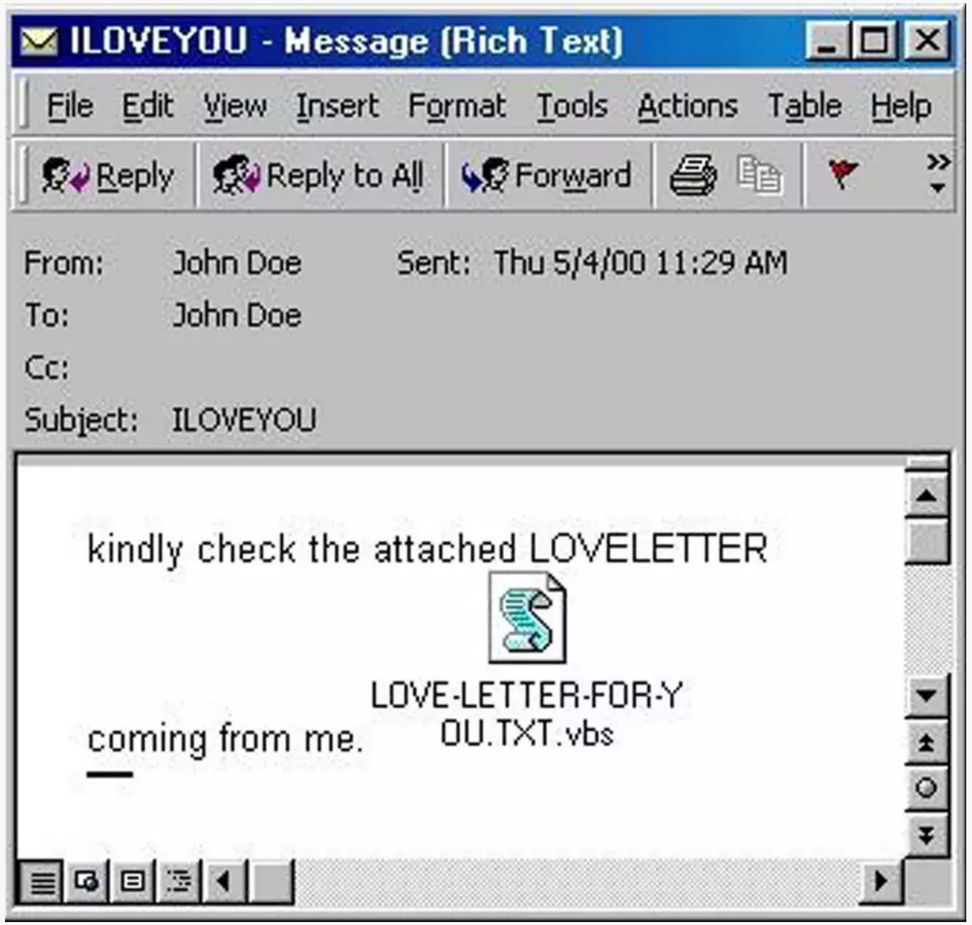
INSTRUCTOR
Dr. Ammar Odeh

PDF Malware Analysis

Done by:
Sara Abbas **20190716**
Maisa Suleiman **20190142**
Kholoud Qubbaj **20190116**
Abdelrahman Shabaneh **20190546**

Introduction

On 2000 May 5th, a malware called ILOVEYOU disrupted the world, once the text file is opened the full file extension is revealed, it's a visual basic script that uses an exploit in early computer systems that allow it to run system code simply by being opened, once the damage is done the worm enters outlook scans for emails then sends itself using the originator's email address, in just about 10 days, the estimated damages are said to exceed 7.8$ Billion in wipe files and another 15$ Billion for cleanup.



System Description

The proposed system is a machine learning-based approach to analyzing PDF files by extracting their physical visual features, then classifying the PDF files as malicious or benign.

The system consists of 5 main components
(1) **User input:** suspected PDF file along with the chosen machine learning model
(2) **Data pre-processing:** getting image dimensions, removing noise & transforming the image to byteplot
(3) **Feature engineering:** extracting and selecting the image features of the PDF file
(4) **Classification:** finally inputting these features into a classifier
(5) **User interface:** result display- outputting the result to the user.

The system consists of three main subsystems divided based on the role chosen by the user:

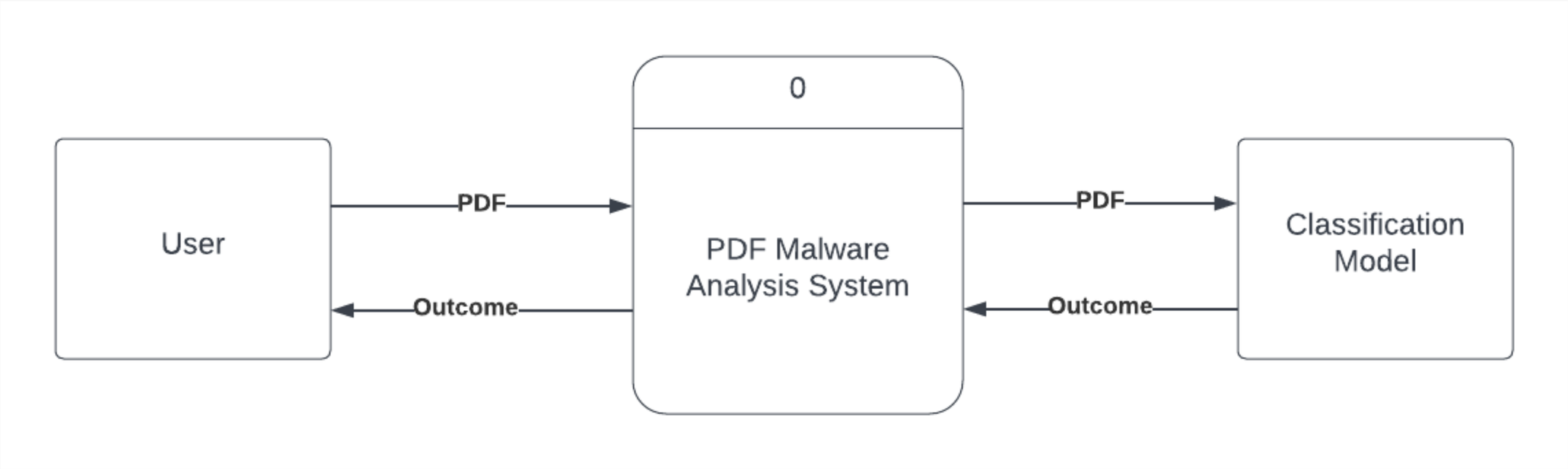
- Standard User
- Researcher User
- Data Engineer

Image Analysis in PDF malware detection

This paper focuses on image properties that could lead to finding abnormal image areas aiding in malware detection. The features consist of **keypoint descriptors** extracted by the SIFT algorithm, usually used for object detection and pattern recognition.

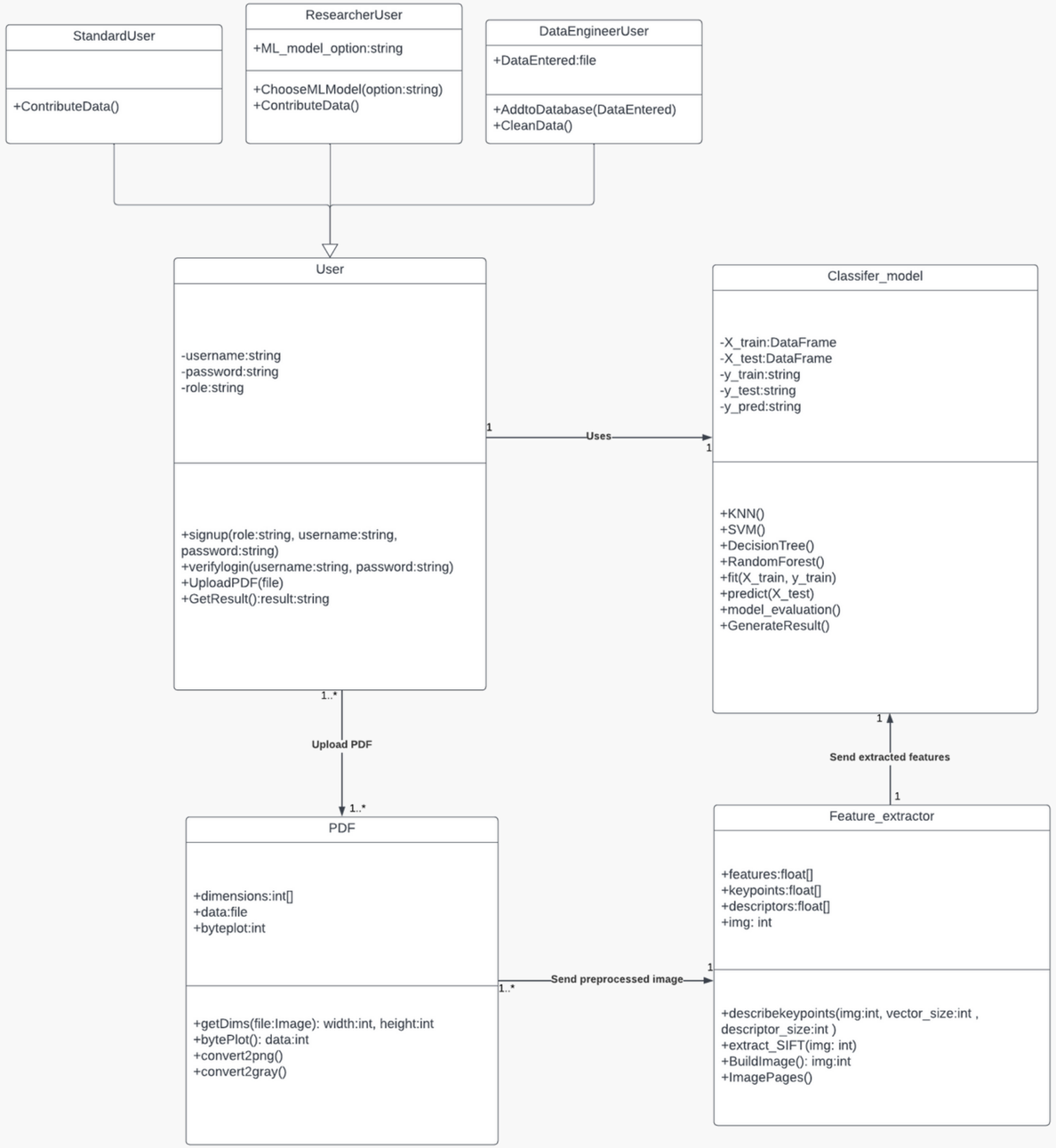
The benefit of taking PDFs as images to perform malware analysis lies in the fact that a lot of PDF malware attacks mimic the structure of a benign PDF file, therefore some visual properties can lead us to a different and more effective way of malware detection.

System Context View

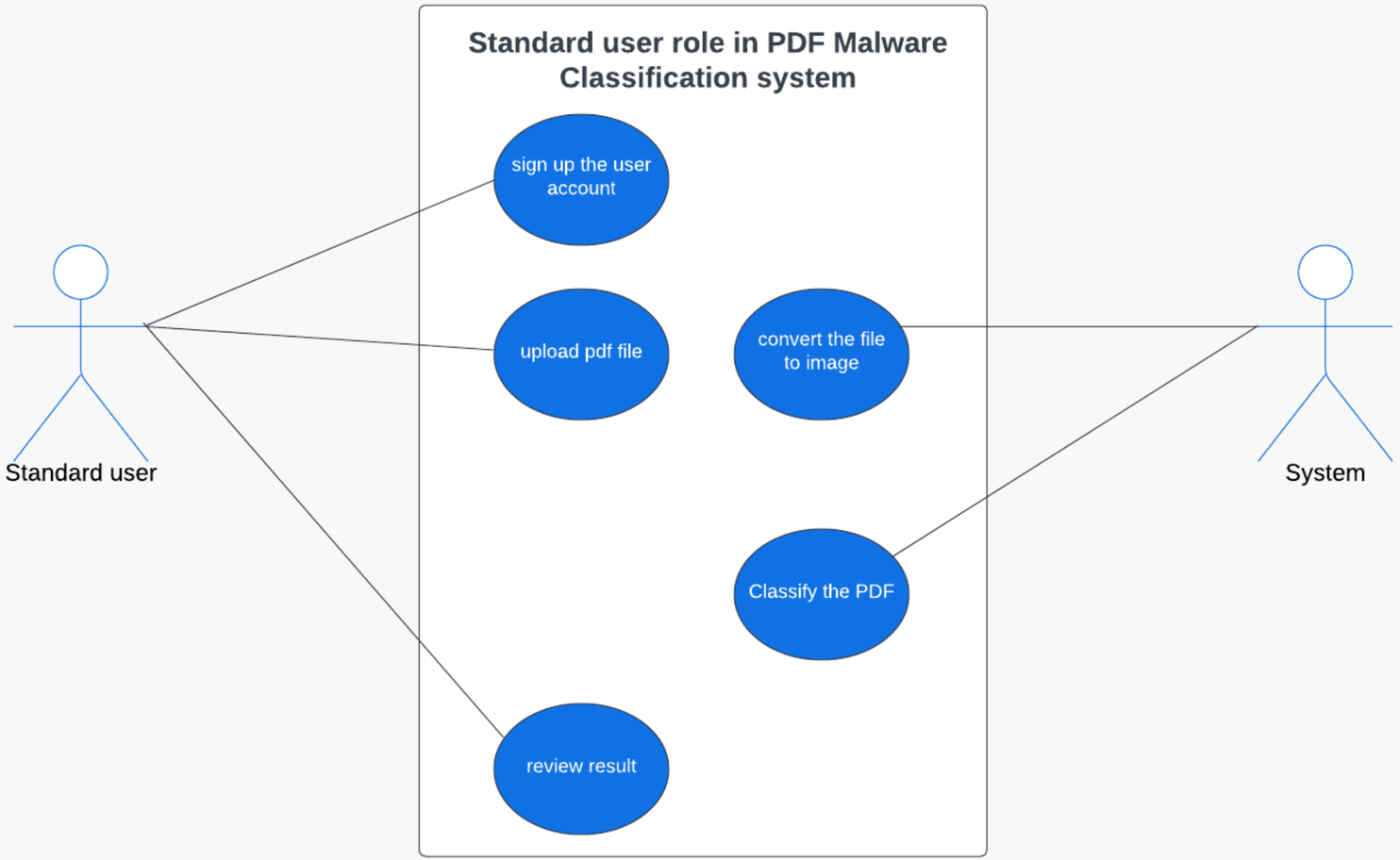


Class Diagram

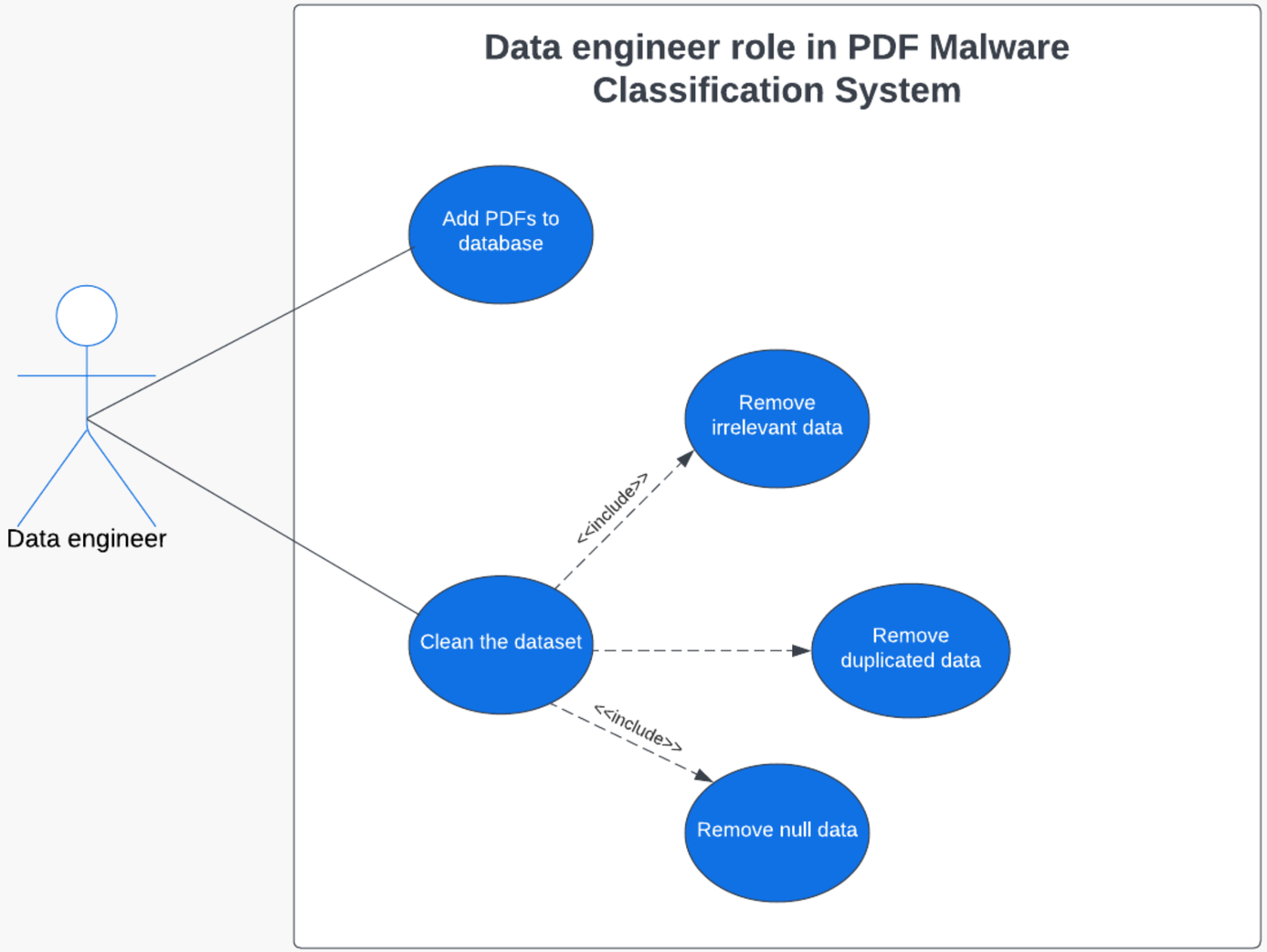
This class diagram contains four classes which are: the user, PDF, Feature_extractor, and the classifier_model. Moreover, it describes the attributes and operations included in each class. It also shows the the realationships between the classes.



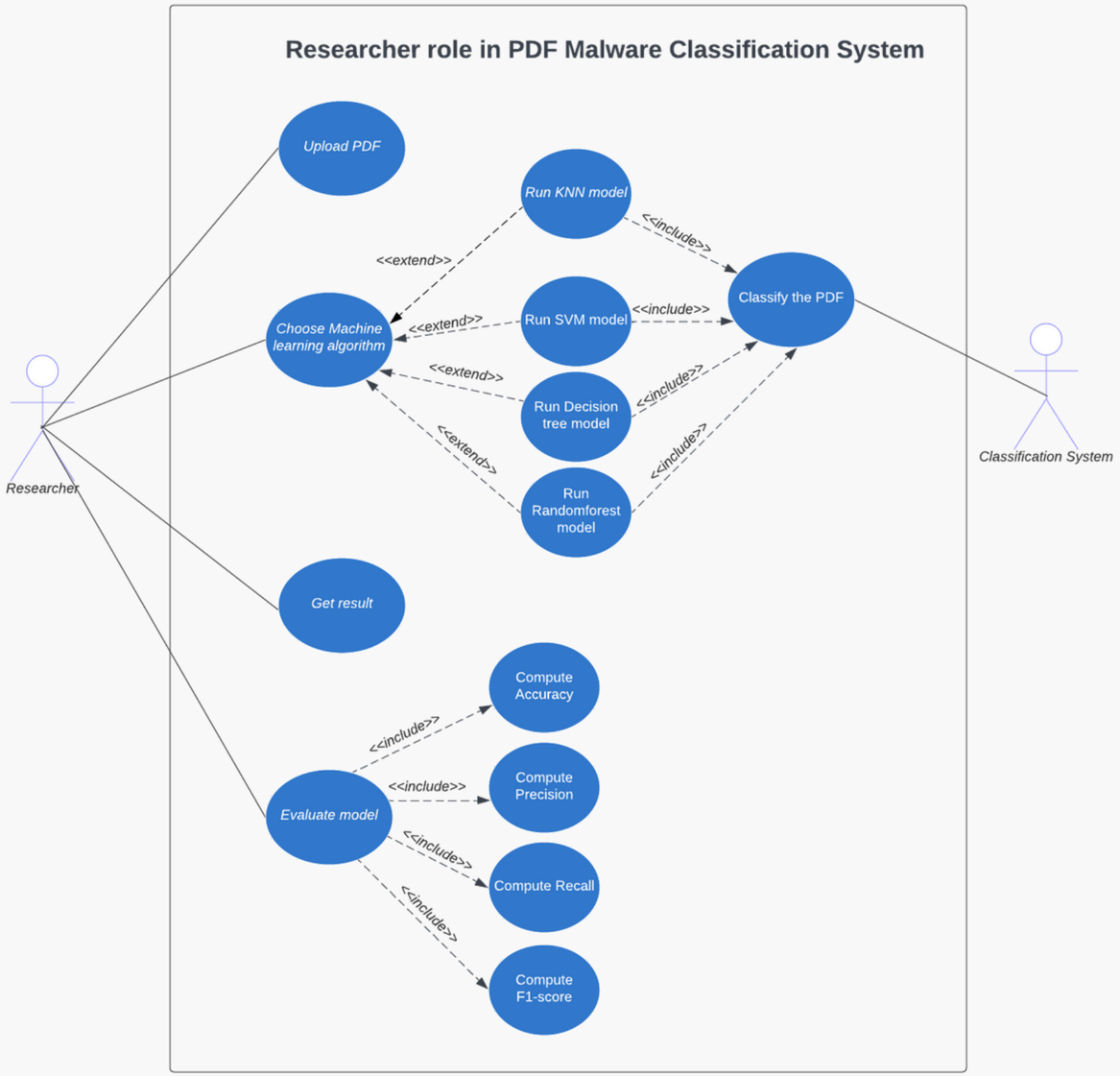
Use case diagram for standard user



Use Case Diagram for data engineer

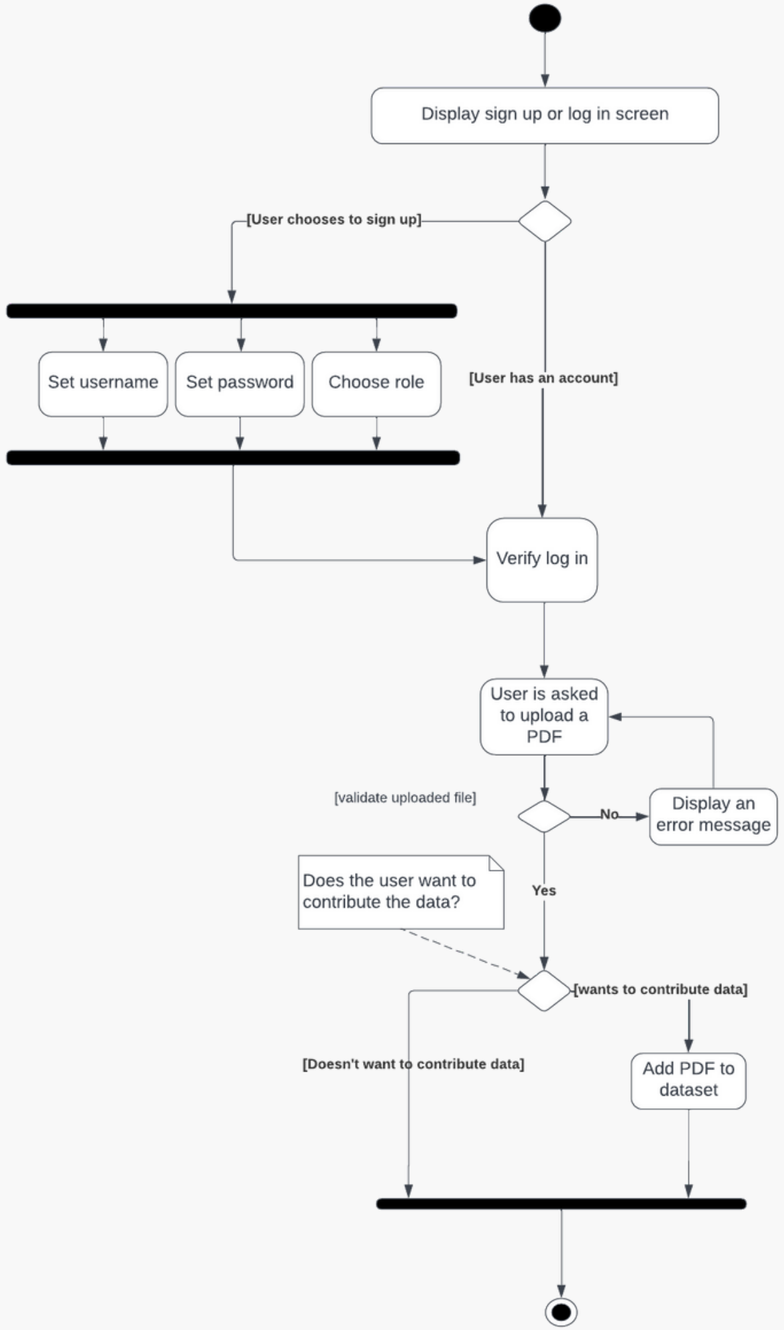


Use Case Diagram for researcher



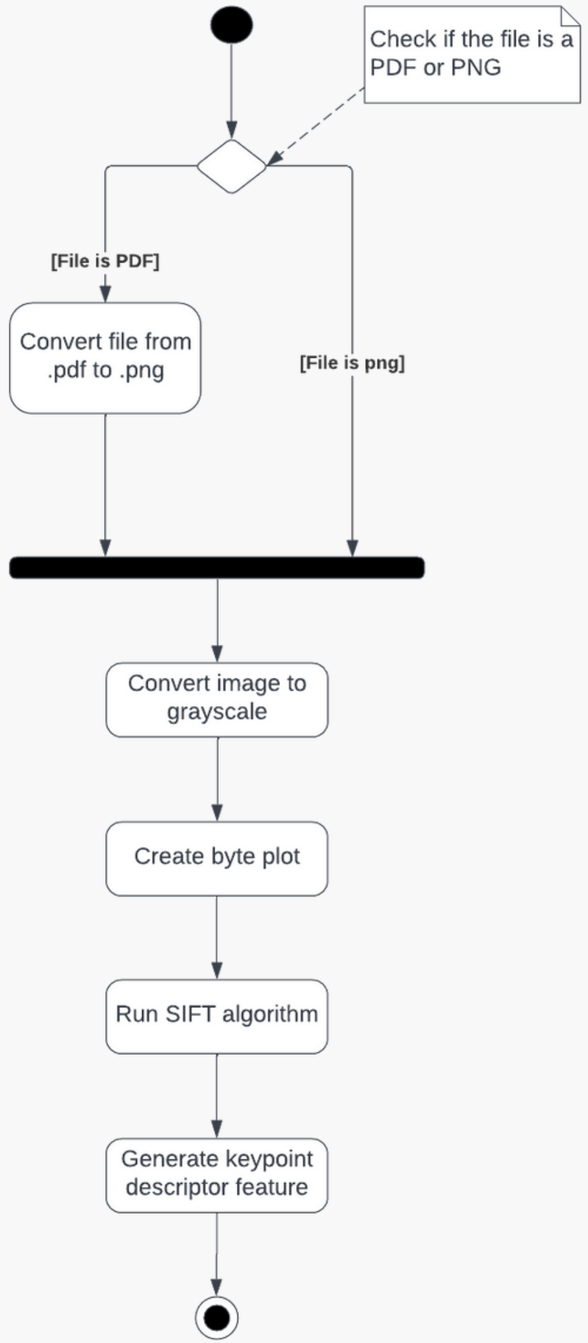
Activity Diagram for sign up/login page in the user interface.

This diagram describes how the user could use the system, the user will login or signup in case he doesn't have an account. Once the user successfully enters the system he will be asked to upload a file with .pdf or .png extension. Later on, they're asked if they want to contribute their data and have it added to the dataset.



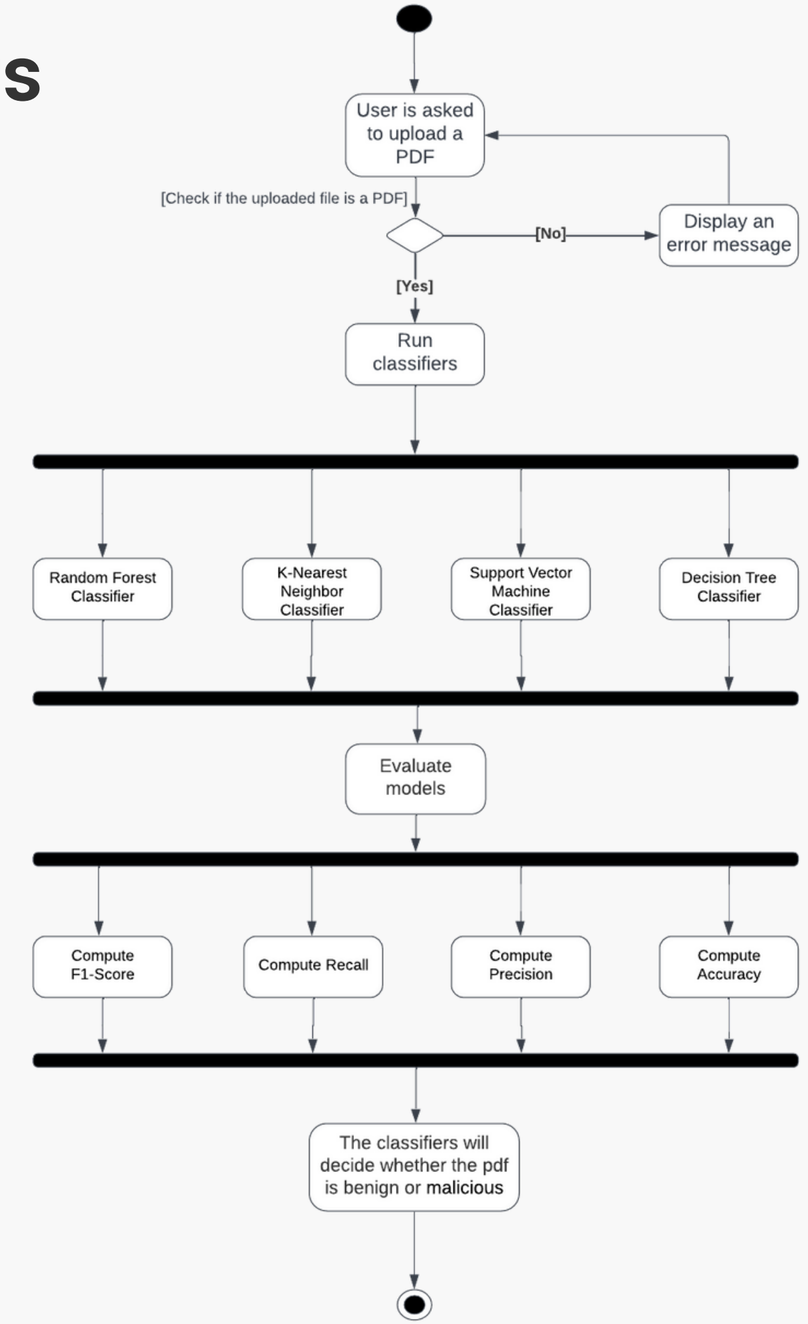
Activity Diagram for image processing of a PDF file.

This activity diagram explains how the PDF files will be converted to images, once all files have a unified form they will be sent to the image preprocessing stage. The image preprocessing stage includes transforming the image to grayscale and then creating a byte plot. Lastly, SIFT Algorithm uses the byte plot generated to create the keypoint descriptor for the features.



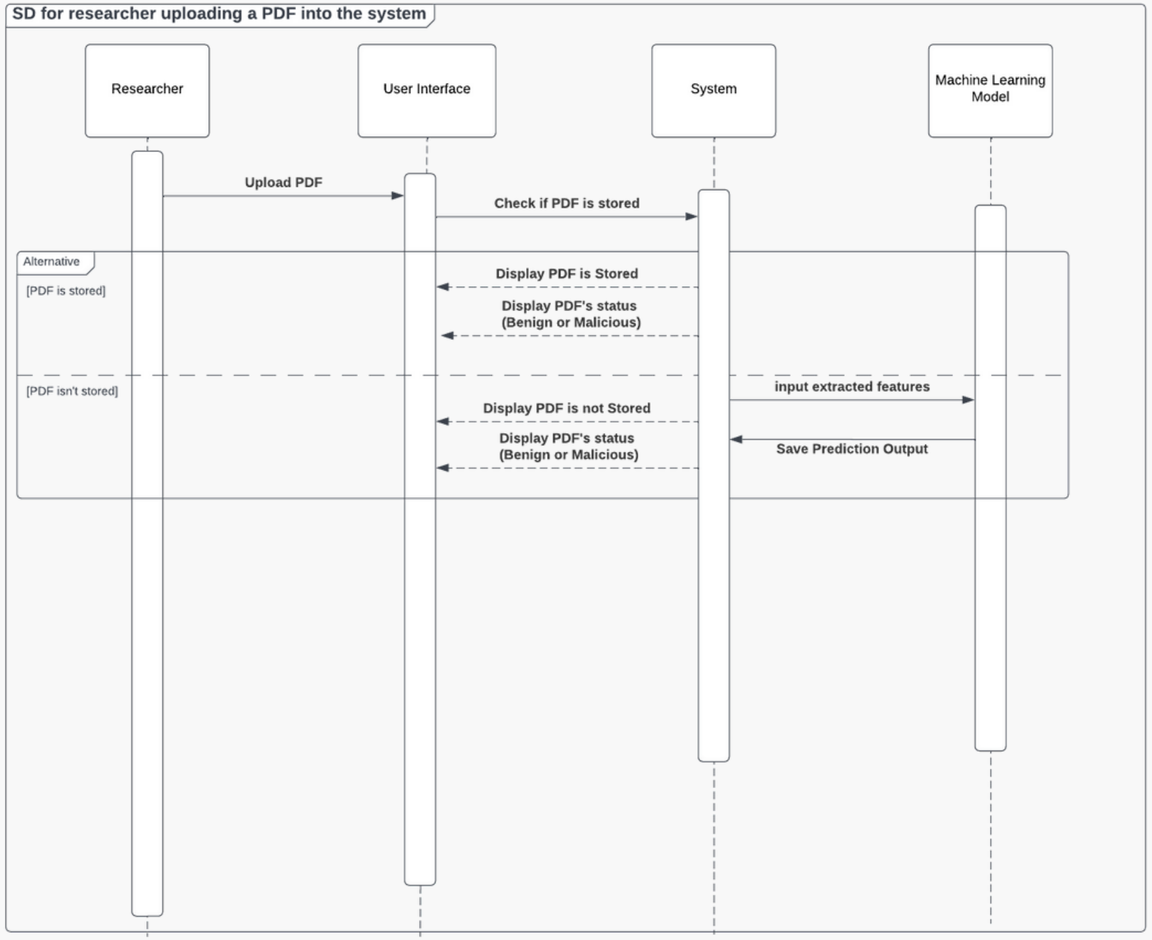
Activity Diagram for ML classifiers when a standard user uploads a PDF.

This activity diagram explains how the PDF files will be converted to images, once all files have a unified form they will be sent to the image preprocessing stage. The image preprocessing stage includes transforming the image to grayscale and then creating a byte plot. Lastly, SIFT Algorithm uses the byte plot generated to create the keypoint descriptor for the features.



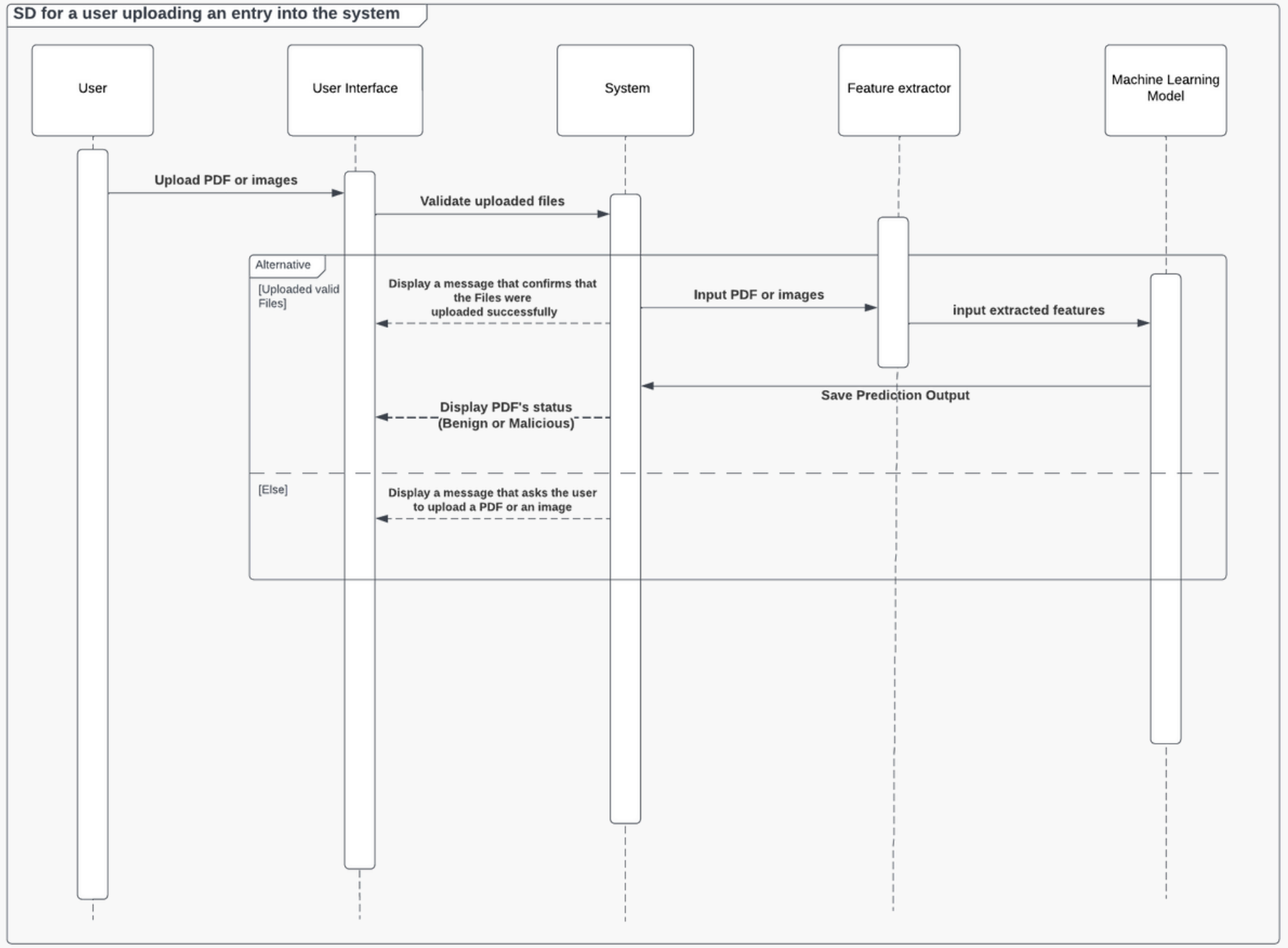
Sequence Diagram for a researcher uploading a PDF into the system.

The following sequence diagram shows the researcher uploading a PDF into the system, our system then checks whether the upload PDF is stored in the system or not, if the file is stored in system's database a message will be displayed telling the researcher that the PDF is stored and it's status (benign or malicious), otherwise the system will display the PDF isn't stored, the system will then extract the PDF features and input it into one of 4 trained machine learning models , the prediction output for the PDF will be stored in the system now, and the prediction output will be displayed for the researcher.



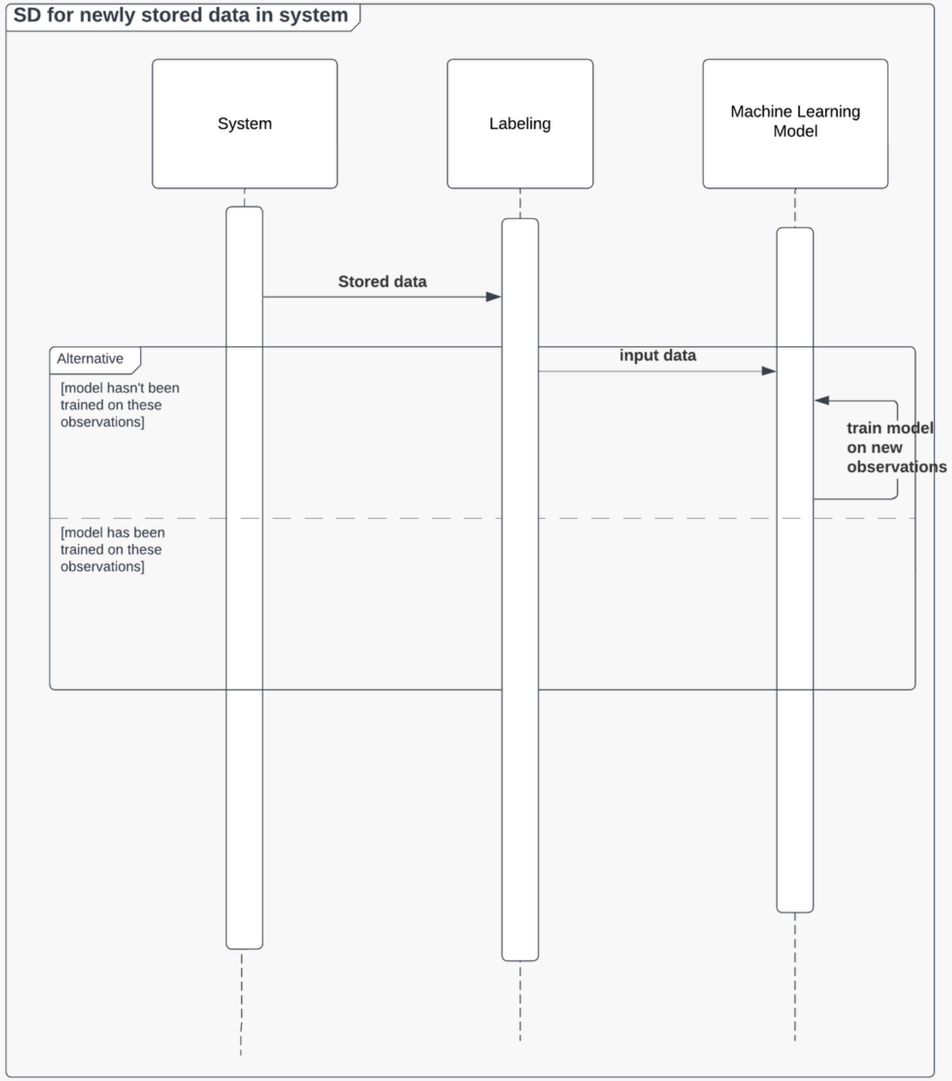
Sequence diagram for a user uploading an entry into the system

The figure shows a sequence diagram for a standard user uploading a PDF or an image into the system, our system then validates the uploaded files, a message confirming that the file was successfully uploaded will be displayed for the user if the uploaded file is valid, otherwise, the system will display a message asking the user to upload a PDF or an image file only, the system will send the uploaded files to the feature extractor, in which the features extracted will then be sent to the trained machine learning model, the predicted output will be stored in the system, and then displayed to the user.



Sequence diagram for newly stored data.

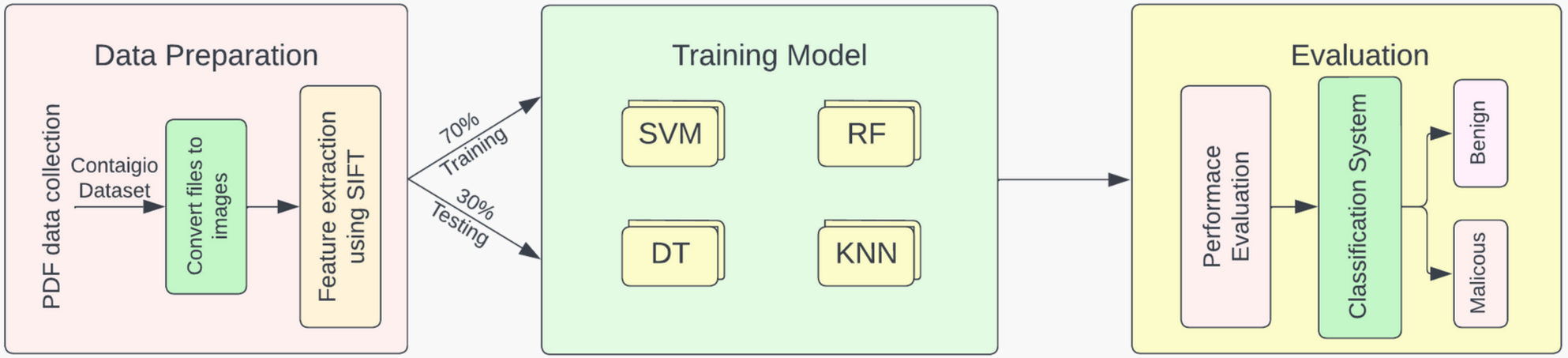
This sequence diagram for retraining the machine learning model on new data points, newly stored data are annotated and labeled by 4 annotators, the machine learning model will be trained on these labeled data points.



Proposed Algorithm

The general system pipeline architecture is made up of 3 main components

(1) Dataset preparation
(2) Model training
(3) Model evaluation

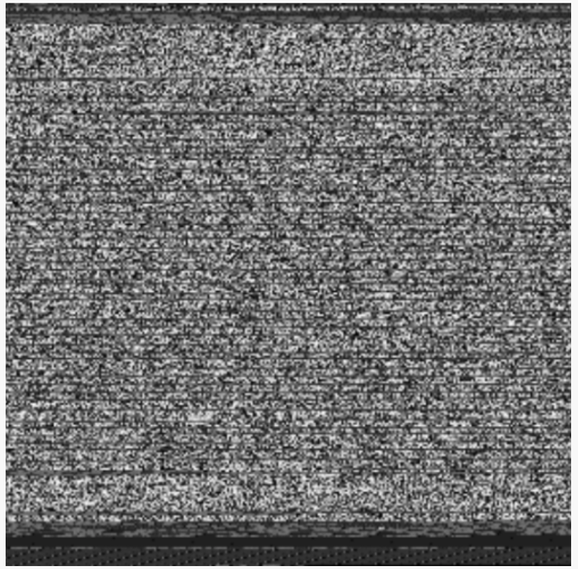


(1) Data preparation & Feature Extraction

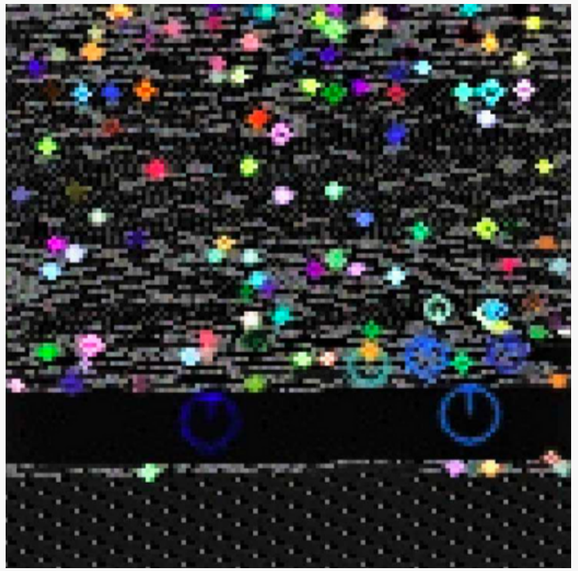
- Changing the PDF files to byte plot images.
- Dataset directory organization
- Feature extraction using SIFT

Byte plot: A visualization for analyzing and interpreting binary data. They are used to detect patterns and trends, and they're also helpful in detecting anomalies which will help us to distinguish malicious from benign files.

SIFT: Algorithm to identify distinctive features such as edges or corners and describe them as feature descriptors. Feature Descriptors are 128-bit vectors that represent the local appearance of the image.



An example of a benign PDF byte plot



Keypoint descriptors of a PDF byte plot

(2) Model training

- **SVM:** uses a hyperplane in the feature space that maximally separates the different classes, unseen data are then classified based on which side of the hyperplane they fall on.

- **Random Forest:** consists of different decision trees. It trains the model on a random data sample, and then when predicting new data points it takes the average of all the decision trees predictions.

- **Decision Tree:** constructs the tree by starting at the root node and then partitioning the data into subsets based on the values of the features. The tree is grown by repeatedly partitioning the data and adding new nodes until some stopping criterion is reached

- **KNN:** prediction for a data point is found by selecting the K nearest neighbors in the training set and taking a majority vote among their class labels

(3) Model Evaluation

- **Accuracy**
- **Precision**
- **Recall**
- **F1 score**

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

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\text{-}score = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

DEMO

Simulation Results

Performance metrics of the four proposed models

Model	Accuracy	Precision	Recall	F1-score
KNN	60.4%	46.7%	95.2%	62.7%
Decision Tree	77.9%	67%	72.2%	69.4%
SVM	95.2%	97%	88.9%	92.8%
Random Forest	89.7%	96.0%	73.7%	83.4%

Comparison with prior work

Accuracy comparison of the proposed algorithm with prior work

Reference	Sample size	Model(s)	Accuracy
[1]	9,000	VGG19	97.3%
[2]	27,000	SVM	95.95%
[3]	10,868	CNN+VGG19	99.08%
[4]	21,146	Random Forest	99.88%
[5]	13,070	SVM	98.88%
Proposed algorithm	15,000	SVM	95.16%

Conclusion and Future work

Our top priority going forward is to obtain higher performance metrics and decrease the false negative rate. This can be done by optimizing the models parameter values, performing cross-validation, doing an ensemble method, or trying new classifiers. We also aim to train the models on a larger dataset from different sources. We would like to build a maintenance system where data from contributive users can be used to retrain our model, to improve the generalizability of our data. We would also like to build a simple user interface to improve the usability of our system. This paper aimed to help not only standard users but researchers too, as it allows the user to choose four different models to compare between.

References

[1] C.-Y. Liu, M.-Y. Chiu, Q.-X. Huang, and H.-M. Sun, “PDF malware detection using visualization and machine learning,” SpringerLink, 14-July-2021. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-81242-3_12.

[2] K. Kancherla and S. Mukkamala, “Image visualization based malware detection | IEEE conference,” IEEE Xplore. [Online]. Available: https://ieeexplore.ieee.org/document/6597204/.

[3] Z. Ren, G. Chen, and W. Lu, “Malware visualization methods based on deep convolutional neural networks - multimedia tools and applications,” SpringerLink, 16-Dec-2019. [Online]. Available: https://link.springer.com/article/10.1007/s11042-019-08310-9.

[4] G. Giacinto and I. Corona, “A pattern recognition system for malicious PDF files detection,” SpringerLink, 01-Jan-2012. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-31537-4_40.

[5] A. Makandar and A. Patrot, “Malware class recognition using image processing techniques,” IEEE Xplore, 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8073489/.