PLAGIARISM DETECTION IN SOURCE CODE AND TEXT BASED ASSIGNMENT SUBMISSIONS INCORPORATING QUANTUM APPROACH

Project Supervisor:
Prof. Dr. Shashidhar Ram Joshi

Outline of topics

- Plagiarism detection in electronic text assignments
- Plagiarism detection in programming assignments
- Quantum approach of classification
- Web application development and system workflow

Plagiarism Detection in electronic text assignments

Types of plagiarism

Verbatim plagiarism

Certain portion of text from the source documents are directly copied into the suspicious document.

Random obfuscation

It is done by performing a sequence of random text operations such as shuffling, adding, deleting, and replacing words or short phrases at random.

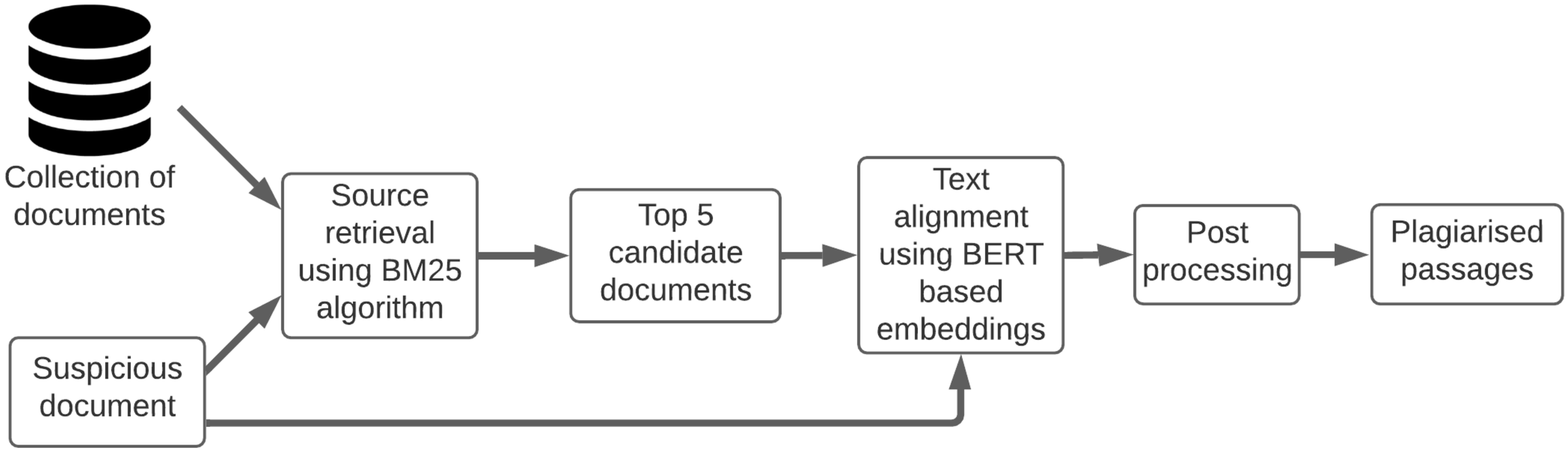
Translation obfuscation

Passage is passed through a sequence of language translators and finally converted to English.

Summary obfuscation

Certain portion of text in the source documents is summarized and added to the suspicious document.

Framework for Text Plagiarism Detection



Text Alignment

Compares source and suspicious documents on the sentence level and form contiguous passages of text

Seeding

Outputs list of tuples (Ssusp, Ssrc) called seeds.

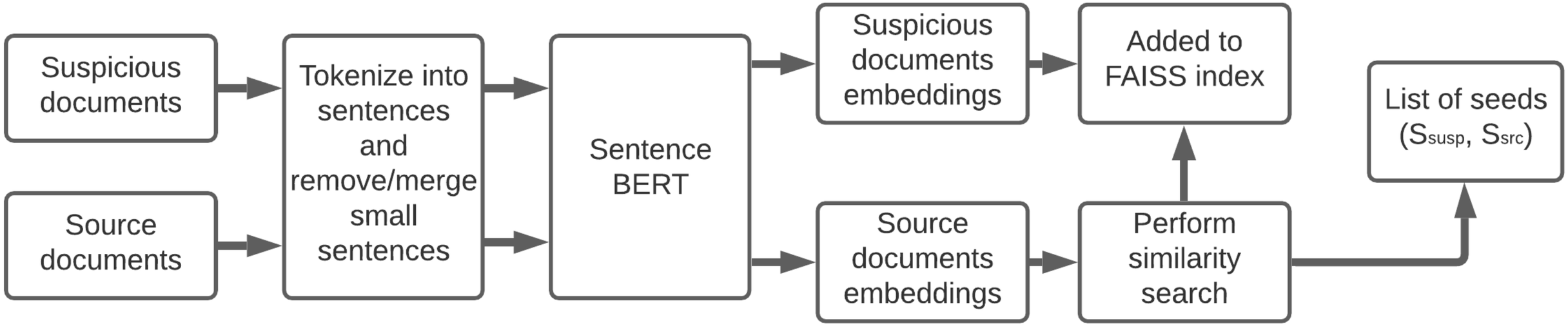
Extension

Forms larger contiguous fragments (clusters) of text which are adjacent to each other

Filtering

Removes overlapping and short plagiarism cases.

Seeding stage of Text Alignment



Result on PAN 2014 training corpus

Our result

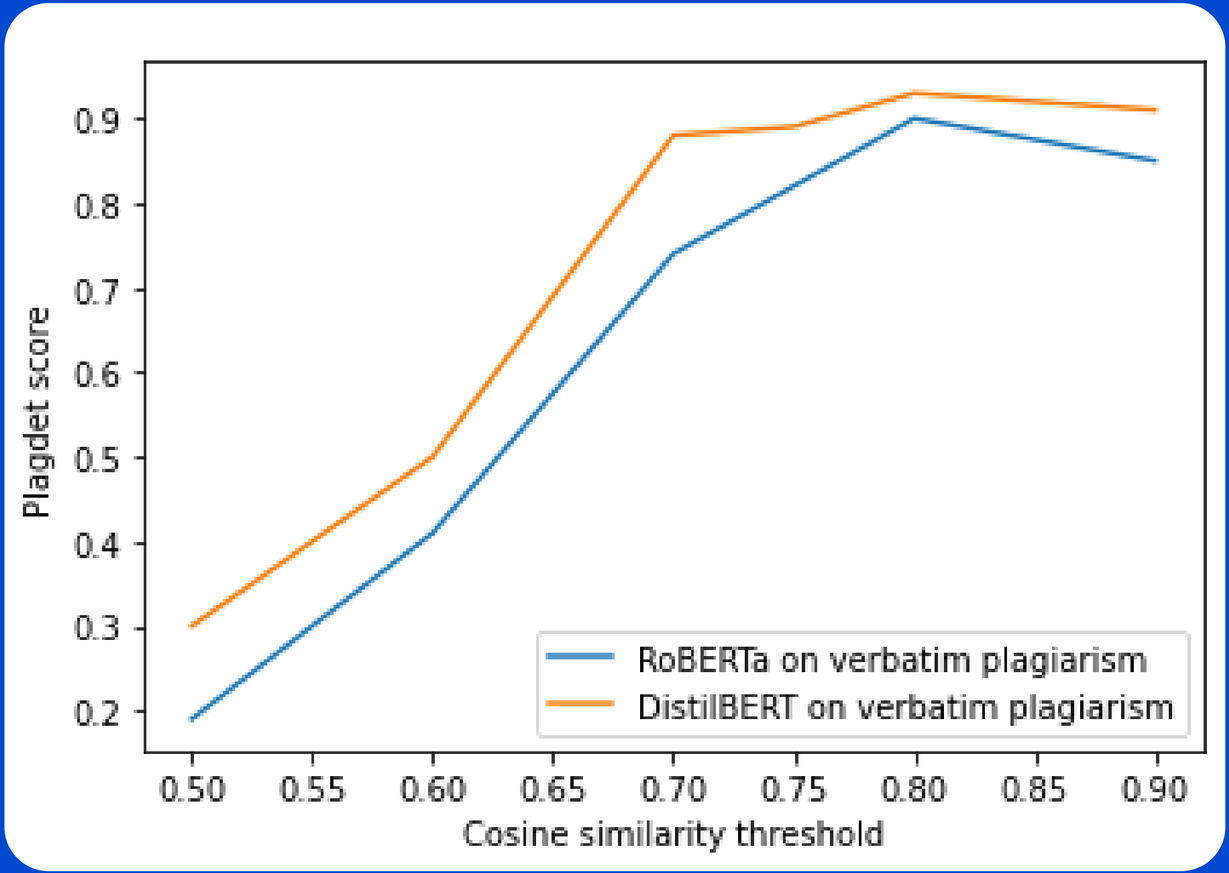
Obfuscation	PlagDet	Precision	Recall	Granularity	Similarity Threshold
None(Verbatim)	0.929	0.947	0.915	1.00246	0.8
Random	0.639	0.778	0.553	1.0164	0.7
Translation	0.75	0.91	0.65	1.019	0.75
Summary	0.676	0.775	0.634	1.044	0.6

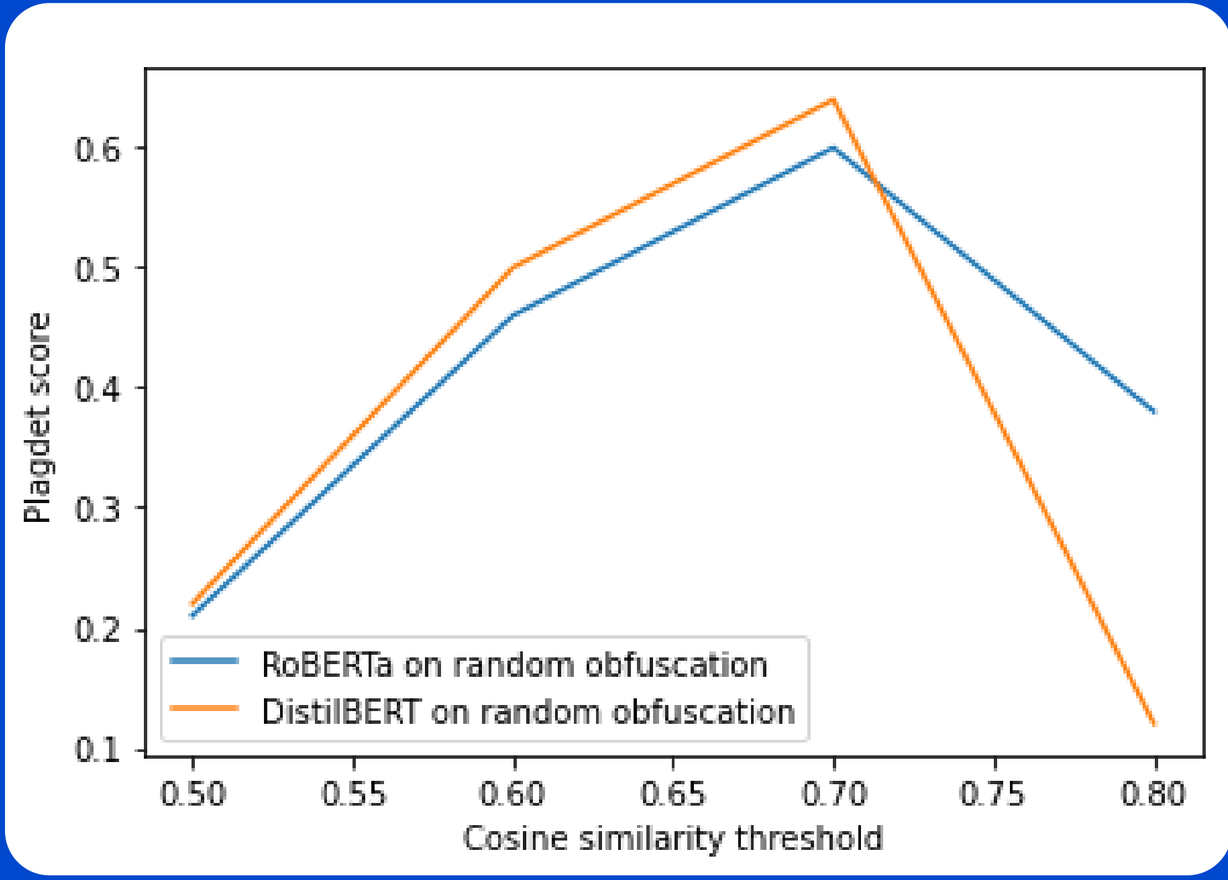
Result of best performing approach in PAN 2014 competition

Obfuscation	PlagDet	Precision	Recall	Granularity
None(Verbatim)	0.8938	0.9782	0.8228	1.0000
Random	0.8886	0.8581	0.9213	1.0000
Translation	0.8839	0.8902	0.8777	1.000
Summary	0.5772	0.4247	0.9941	1.0434

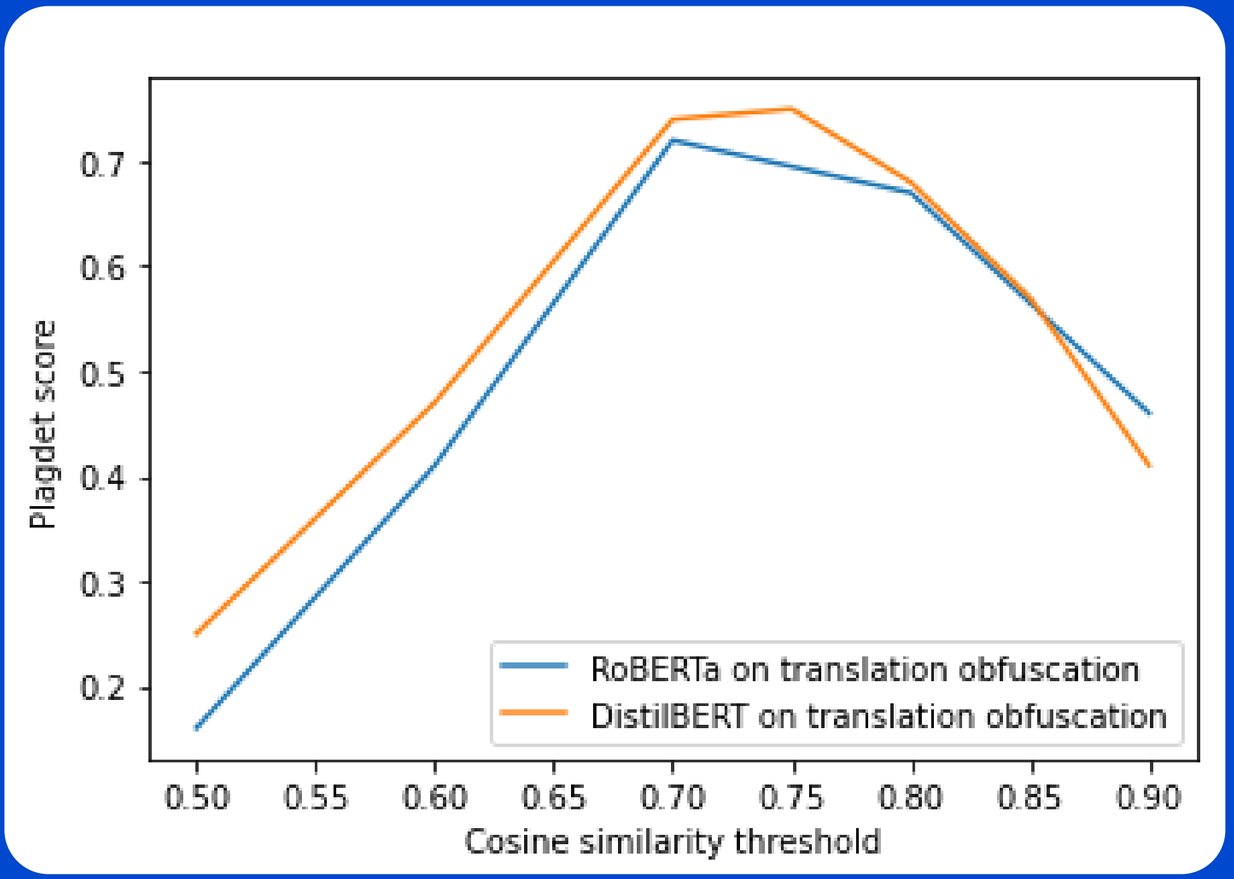
Reference: Potthast, M. et al. (2010). An evaluation framework for plagiarism detection. COLING '10: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, 9971005.

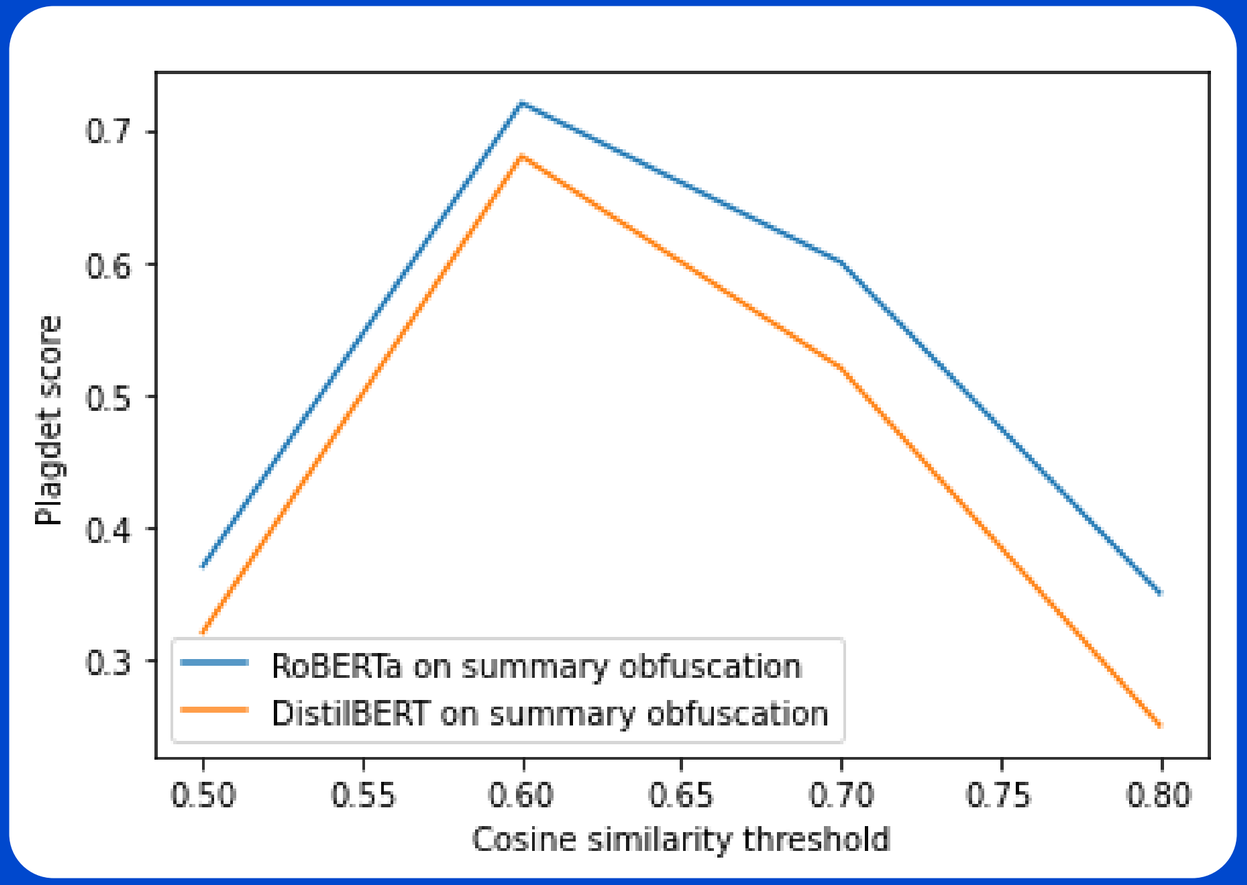
Plagdet score for RoBERTa and distilBERT





Plagdet score for RoBERTa and distilBERT

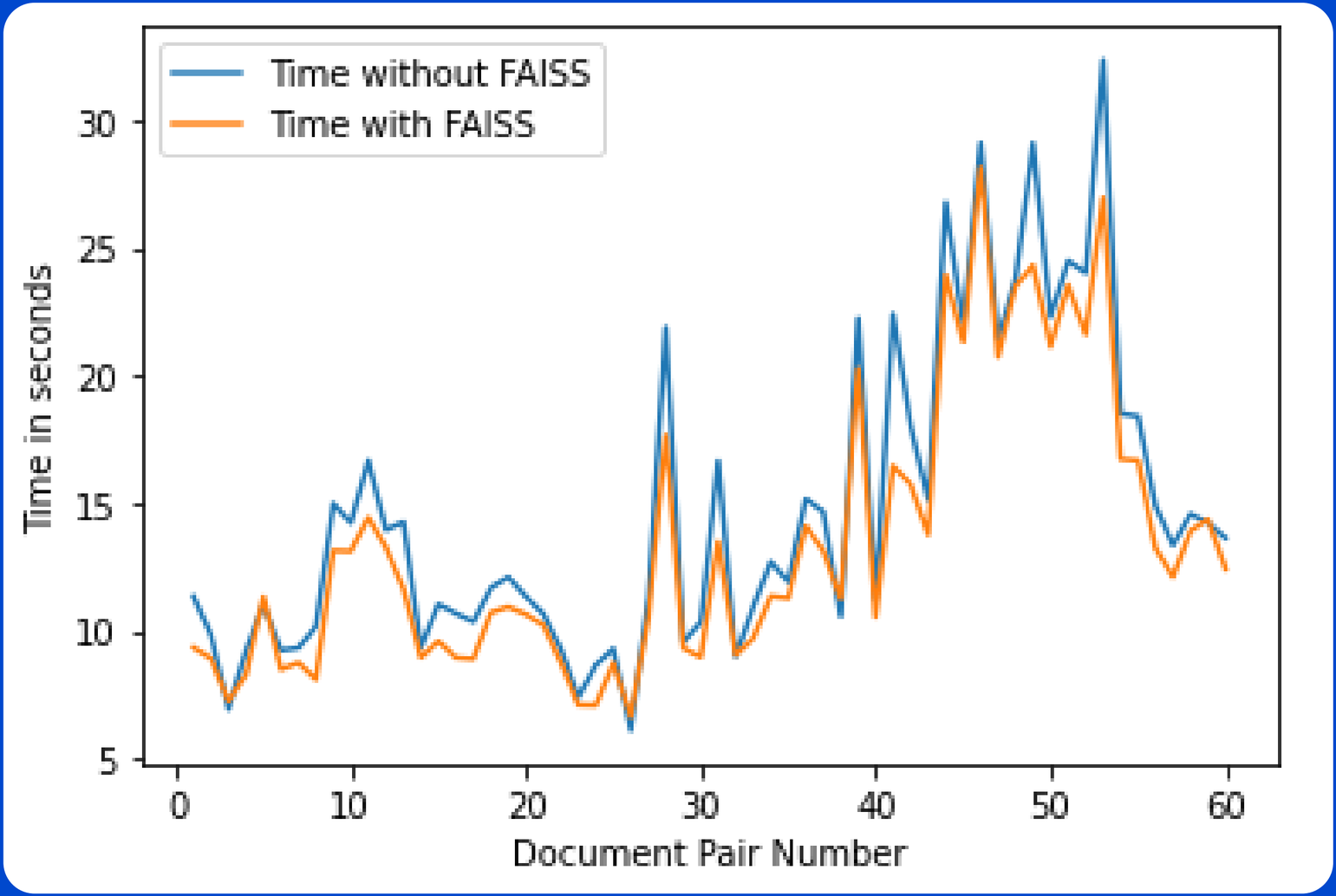




Our result on PAN 2014 test corpus

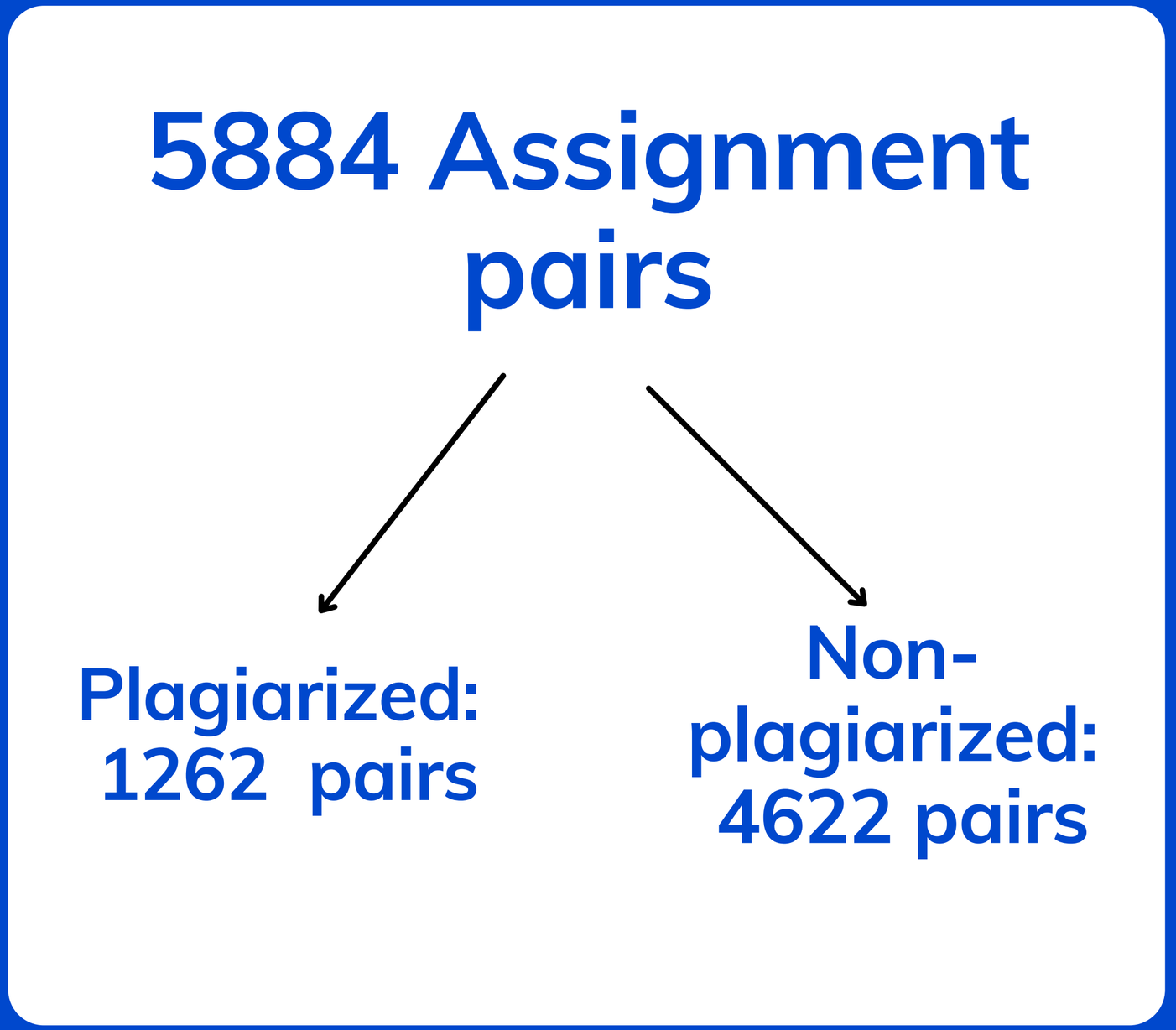
Obfuscation	PlagDet	Precision	Recall	Granularity
None(Verbatim)	0.799	0.693	0.946	1.00253
Random	0.648	0.771	0.568	1.01335
Translation	0.654	0.728	0.603	1.01116
Summary	0.508	0.993	0.36	1.05682

Improvement in running time of algorithm using FAISS



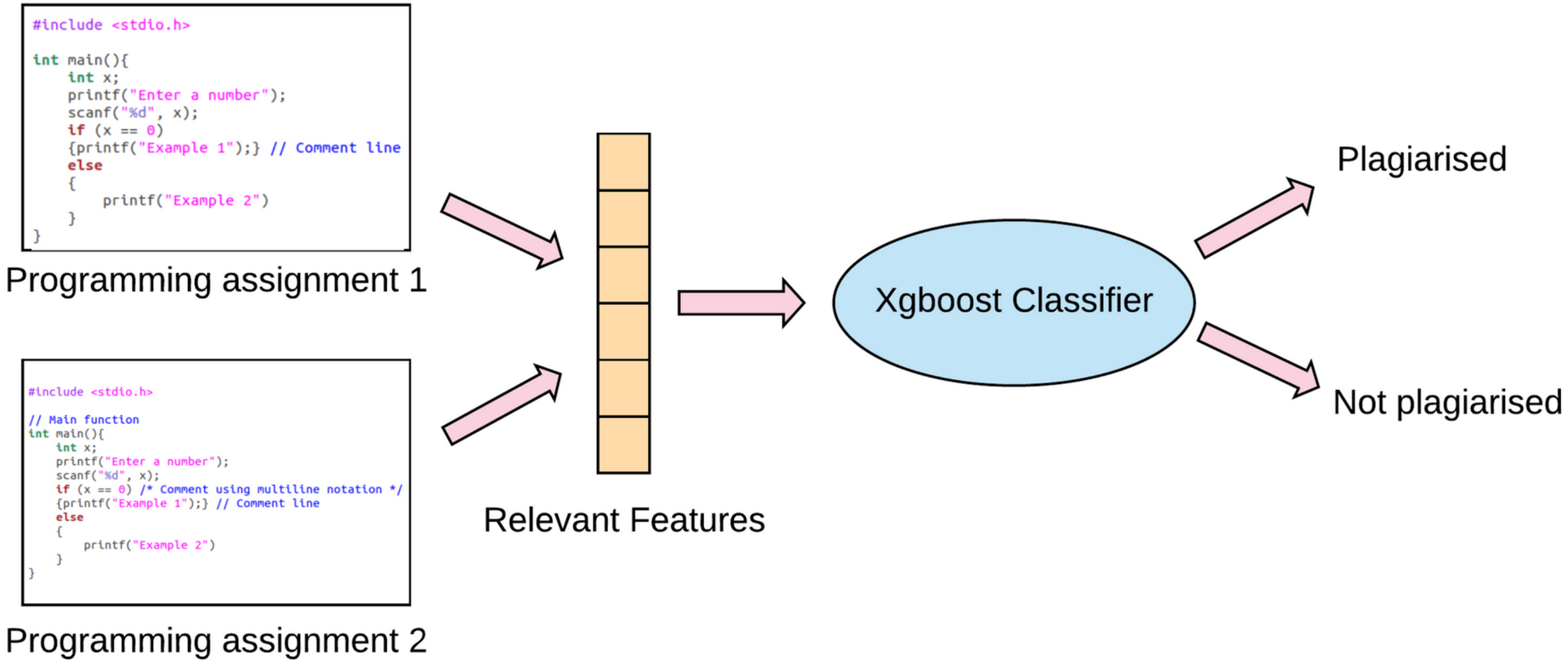
Plagiarism Detection in Programming Assignments

Dataset



Dataset of programming assignments

Our Approach for Source Code Plagiarism Detection



Features Calculated

Similarity score

Score based on Karp-Rabin string matching algorithm and Jaccard similarity.

Code style similarity

- Braces similarity
- Comment similarity
- Spaces and Newline similarity

Categorical value according to similarity score

Number of common lines

Total number of common lines in the assignment pair excluding blank lines.

Number of unused variables

Static code analyzer cppcheck was used to detect unused variables.

Number of unused functions

Static code analyzer cppcheck was used to detect unused functions.

C/C++ code before and after replacing variable names, function names and string

```c
#include <stdio.h>

// Printing Fibonacci series
int main() {
    int i, n, t1 = 0, t2 = 1, nextTerm;
    printf("Enter the number of terms: ");
    scanf("%d", &n);
    printf("Fibonacci Series: ");

    for (i = 1; i <= n; ++i) {
        printf("%d, ", t1);
        nextTerm = t1 + t2;
        t1 = t2;
        t2 = nextTerm;
    }

    return 0;
}
```

```c
int F() {
    int N, N, N = 0, N = 1, N;
    printf(SSS);
    scanf(SSS,&N);
    printf(SSS);

    for (N = 1; N <= N; ++N) {
        printf(SSS, N);
        N = N + N;
        N = N;
        N=N;
    }

    return0;
}
```

An example of braces and comment notation

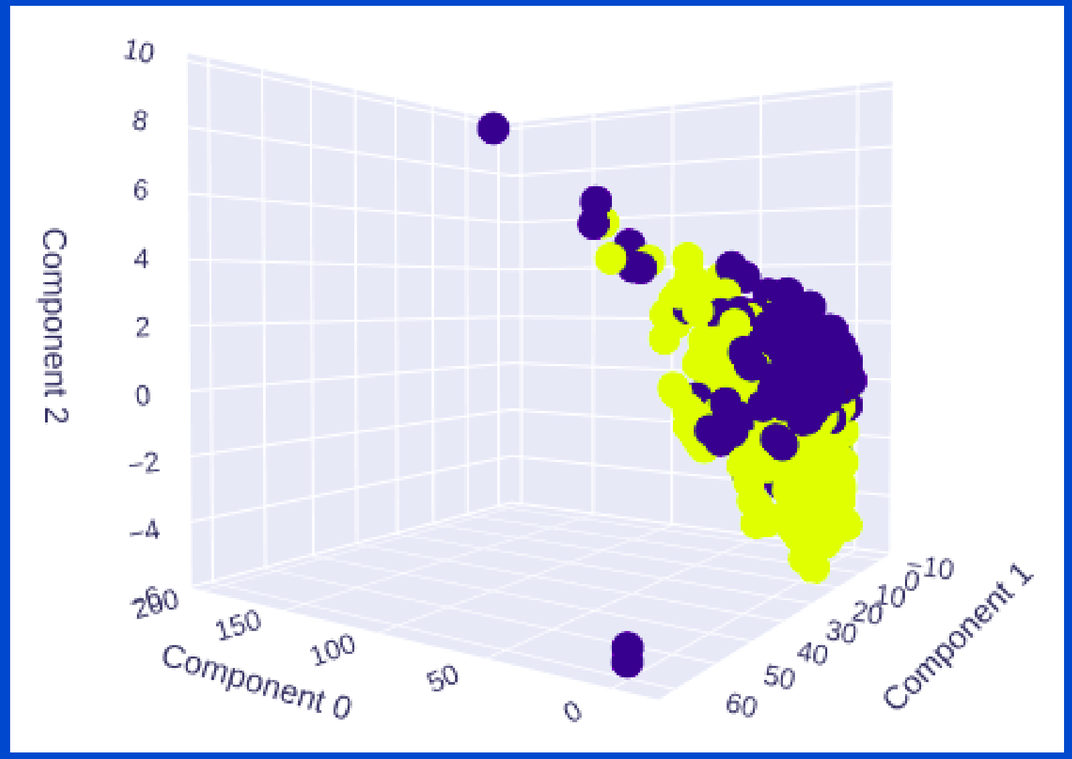
```c
#include <stdio.h>

// Main function
int main(){
    int x;
    printf("Enter a number");
    scanf("%d", x);
    if (x == 0) /* Comment using multiline notation */
    {printf("Example 1");} // Comment line
    else
    {
        printf("Example 2")
    }
}
```

Braces notation of above code is {2{1}3{4}4}4.

Comment notation of above code is S1M3S2.

Visualization of Source Code Features after PCA



Results

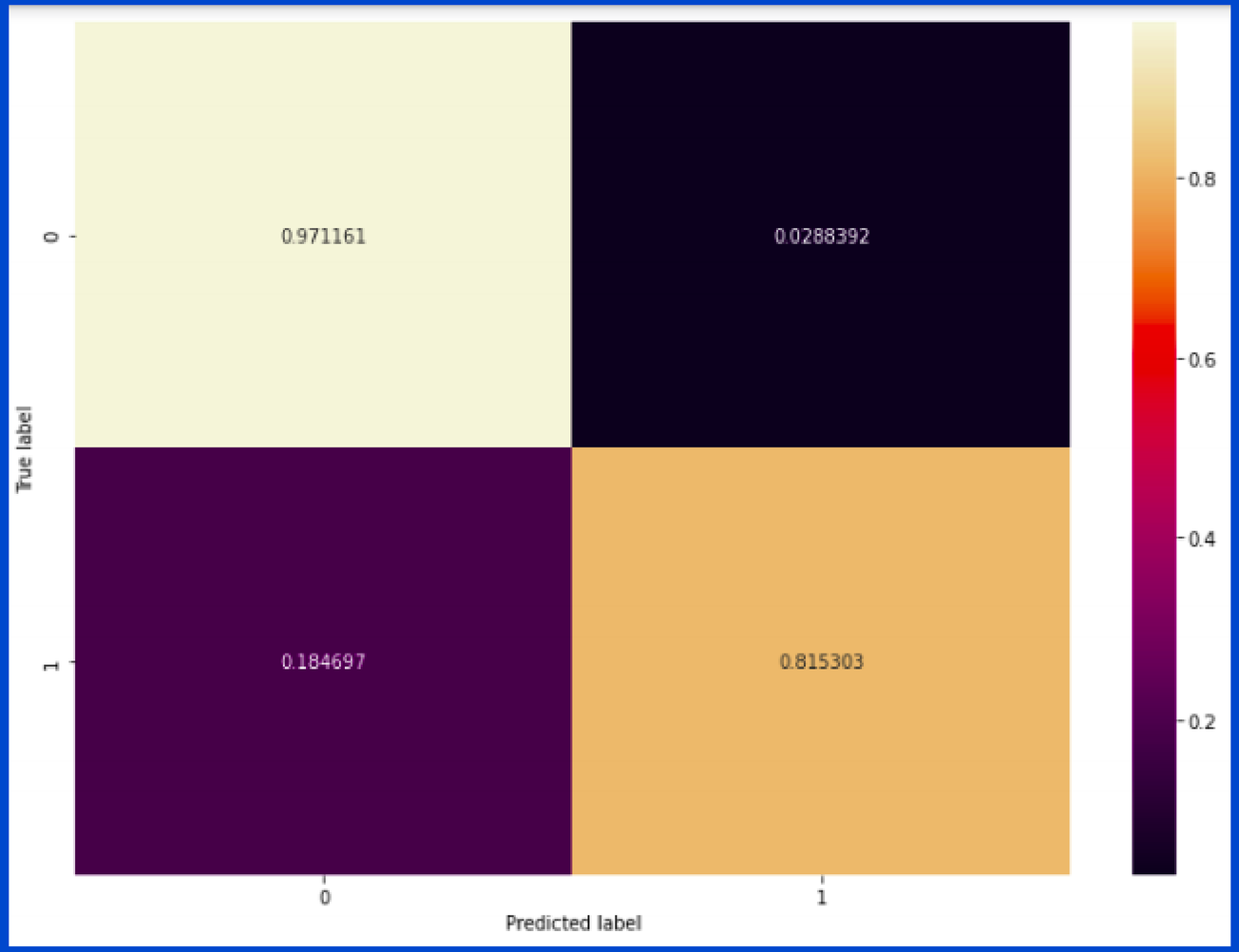
Xgboost

Class Label	Precision	Recall	F1-score	Accuracy
Not plagiarised	0.95	0.97	0.96	0.93
Plagiarised	0.89	0.82	0.85	

SVM

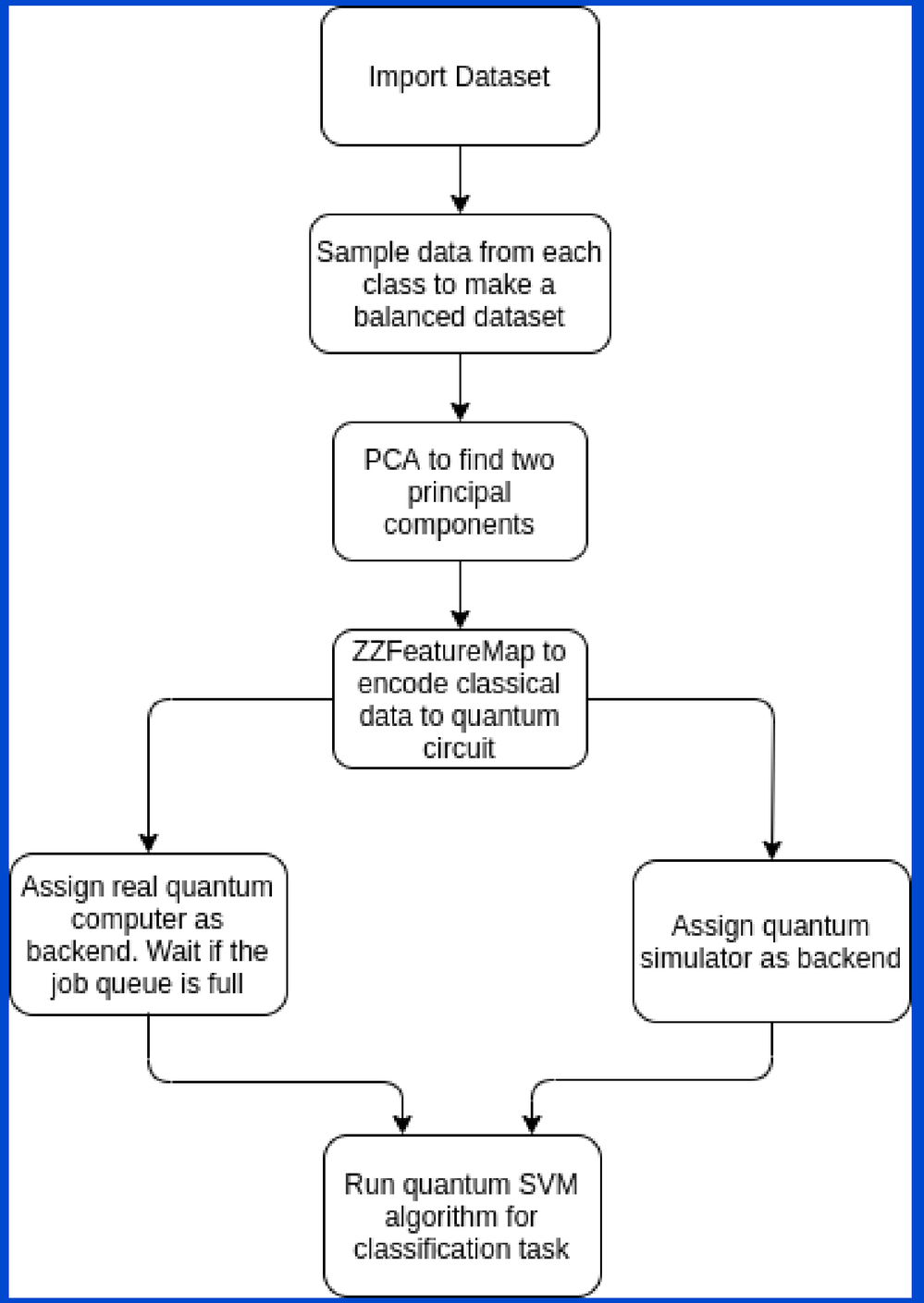
Class Label	Precision	Recall	F1-score	Accuracy
Not plagiarised	0.91	0.97	0.94	0.90
Plagiarised	0.87	0.66	0.75	

Confusion Matrix Normalized by the True labels



Quantum Approach of Classification

System overview of quantum based SVM classification



Results

Results obtained from Quantum Based SVM model

Quantum SVM

Accuracy(testset) : 75 per cent

Run Time : 42.8 seconds

Classical SVM

Accuracy(testset) : 70 per cent

Run Time : 4.2 seconds

Inferences from Results

Explanation on why we got such a surprising result as we expected the run time for quantum based model to be less

Simulation of a quantum computer on classical

- **Increased space complexity**

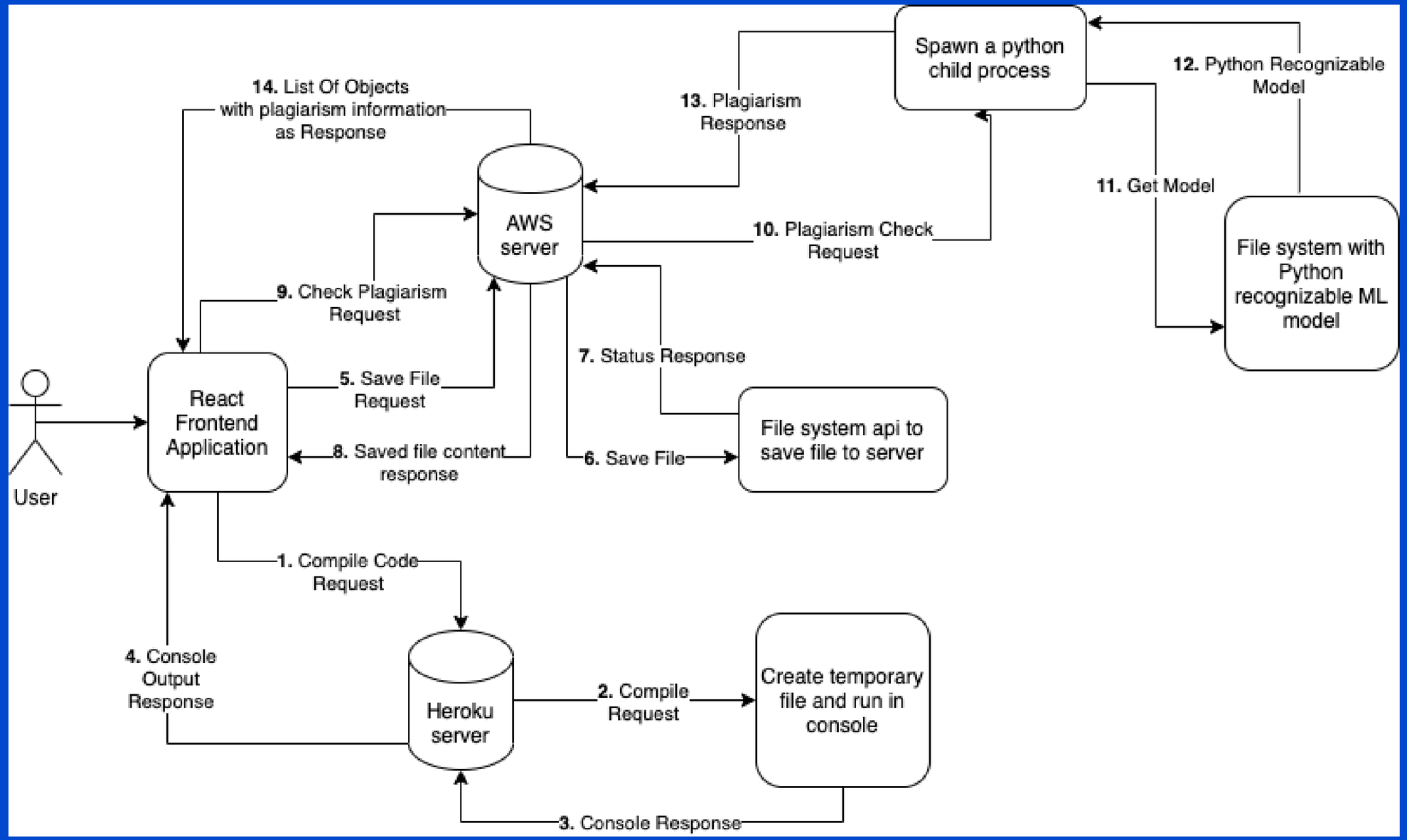
  Simulating an n-bit quantum computer requires to store about 2^n bits of information every instant

- **Increased time complexity**

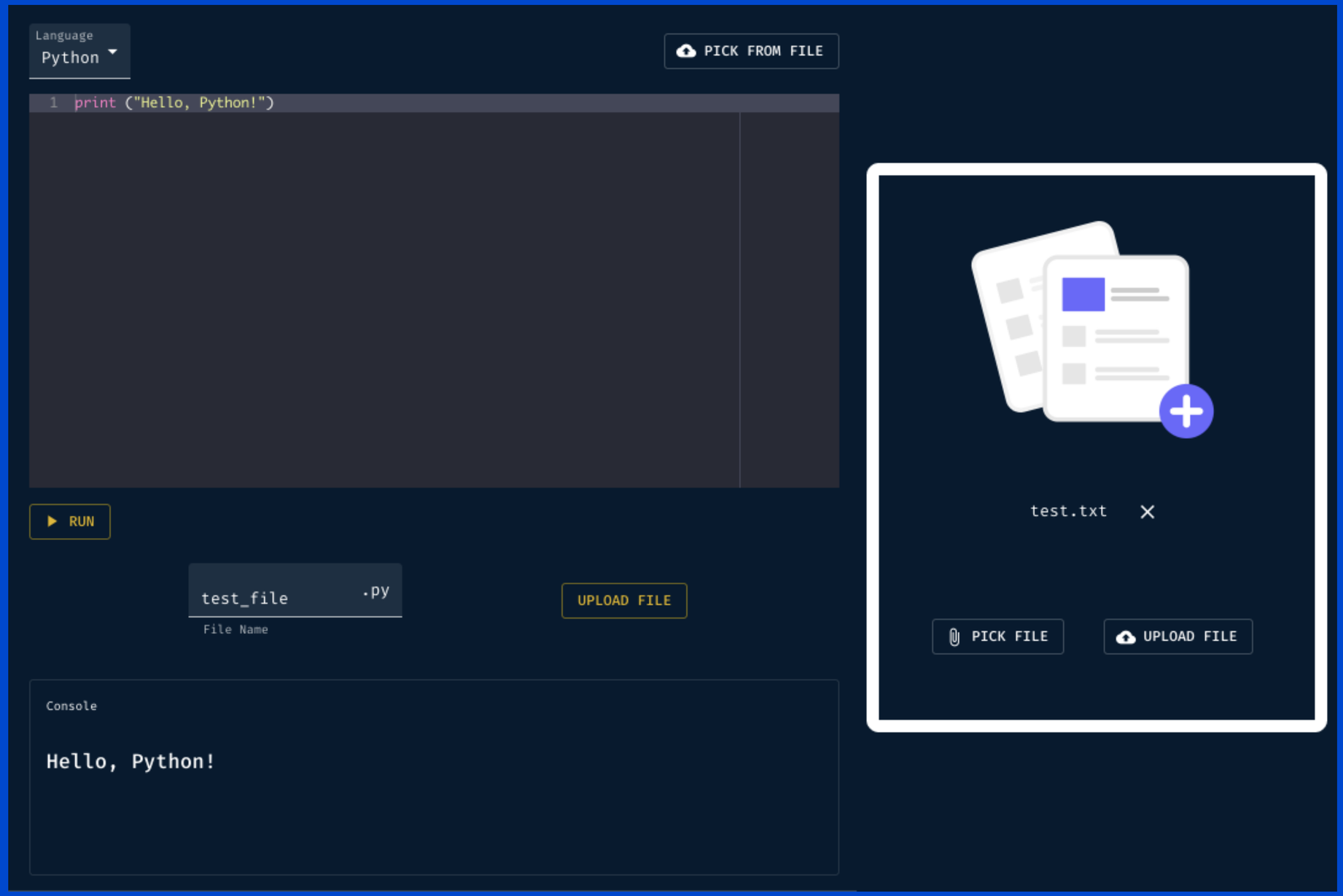
  Implementation of quantum gates on classical computer takes huge time resources.

System Design

Overview of the system



Frontend


Language
Python
PICK FROM FILE
1 print ("Hello, Python!")
RUN
test_file
.py
File Name
UPLOAD FILE
Console
Hello, Python!
test.txt
PICK FILE
UPLOAD FILE

Sample Response for /checkCodePlag

```
{
  "1": ["fib.c", "fibb.c"],
  "2": ["fib.c", "fibbo.c"],
  "3": ["fib.c", "fibbon.c"]
}
```

Sample Response for /checkTextPlag


```
{
  "source_retrieval": {
    "a.txt": ["b.txt"],
    "b.txt": ["a.txt"]
  },
  "final_output": {
    "1": {
      "a.txt": [0, 1219],
      "b.txt": [0, 1219]
    }
  }
}
```

Tech Stack for web application

React
for frontend

Firebase
for frontend hosting

NodeJS
for backend

Heroku
for compiler api hosting

AWS
for plagiarism api hosting

Team Members

Anish Dulal

Bibek Timsina

Mitesh Pandey

Nishesh Awale