GROUP NO: 02

Project Report

on

Automatic Text Categorization of Word Problems

SUBMITTED BY

SURAJ KUMAR, B120084330

PRANAV KANADE, B120084257

SANKET DESHMUKH, B120084228

GAURAV SHUKLA, B120084237

Under the guidance of

Mrs. Shanti K. Guru

In partial fulfillment of the requirements for

Bachelors Degree in Computer Engineering of

SAVITRIBAI PHULE PUNE UNIVERSITY

[2016-2017]

Department of Computer Engineering

D. Y. PATIL COLLEGE OF ENGINEERING

Akurdi, PUNE-411044.

D. Y. Patil College of Engineering

Akurdi, Pune-411044

Department of Computer Engineering

CERTIFICATE

This is to certify that **Mr. Suraj Kumar, Mr. Pranav Kanade, Mr. Sanket Deshmukh and Mr. Gaurav Shukla** have satisfactorily completed the project work entitled **”Automatic Text Categorization of Word Problems”** which is a bonafide work carried out by him under the supervision of **Mrs. Shanti K. Guru** and it is approved for the partial fulfillment of requirement of Savitribai Phule Pune University, for the award of the degree of Bachelors of Engineering (Computer Engg.) for the academic year 2016-17 Sem-I.

Mrs. Shanti K. Guru	Dr. Mrs. Neeta Deshpande
(Project Guide)	HOD (Computer)

Place: Pune

Date:

ACKNOWLEDGEMENT

With immense pleasure, we present the project report as part of the curriculum of the B.E. Computer Engineering. We wish to thank and express deep sense of gratitude to our guide Prof. Mrs. Shanti K. Guru for her consistent guidance, inspiration and sympathetic attitude throughout the project work. Without her this project could not have even been imagined. This project would not have been feasible without encouragement and guidance of our Project Co-ordinator, Mrs. Shanti K. Guru. We are heavily indebted to her. She patiently discussed ideas with us and gave indispensible suggestions.

We are grateful to Dr. Mrs. Neeta Deshpande, Head of Department of Computer Engg., D.Y.P.C.O.E. for always being ready to help with the most diverse problems that we have encountered along the way. We express our sincere thanks to all our staff and colleagues who have helped us directly or indirectly in completing this project.

Last but not the least, We thank Persistent Systems Ltd. who sponsored this project and had immense faith in us.

SURAJ KUMAR	B120084330
PRANAV KANADE	B120084257
SANKET DESHMUKH	B120084228
GAURAV SHUKLA	B120084237

ABSTRACT

Automatic Mathematical Word Problem Solver serves two purposes:First is in Intelligent Tutoring Systems and Second is in Cognitive Science to study way of thinking to solve these problems in children. To solve a MWP, First we need to classify the given problem into either Joint and Seperate problem or Part-Part-Whole problem or Compare problem. After clssification is done we need to recognize the function of each sentence and extract information from them. This extracted feature is finally used to form equations and then equations are solved to obtain the solution.So the proposed system consists of following main modules: Classification, Information Extraction, Equation Generator and Equation Solver.

Contents

List of Figures

1 Problem Definition

Design a system which accepts an algebric word problem as an input and classifies it into pre-defined domain (e.g. Part-Part-Whole, Join-Seperate, Comparison, Equal Groups, Multiplicative-Compare etc.) using NLP (Information Extraction) and Machine Learning Techniques. Also generting equations for these problems and solving them automatically.

1.1 Problem Outcome

The automatic problem solver defines following outcomes:

1. Assistance Tools for students to solve algebric word problems
2. To study thought process while solving a algebric word problem
3. To assist in teaching as Intelligent Tutoring System

1.2 Motivation

While participating for ACM ICPC 2015, we did not get much clue about how to proceed to solve a algorithmic problem. What approach to follow and which methods to use. So we thought to make a system which could automatically suggest data structures and algorithms for the problem. But the problem here was the topic was too vast to start with. So we thought to apply the same for the basic algebric problems and thus in this way we got motivated for the proposed system.

2 Literature Survey

2.1 Classification

Classification[4] predicts examples into given set of categories.

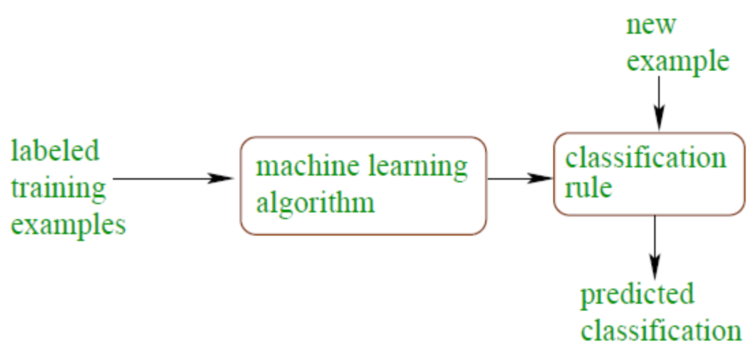


Figure 1: Classification

Examples of Classification Problems:

1. text categorization (e.g., spam filtering)
2. fraud detection
3. optical character recognition
4. machine vision (e.g., face detection)
5. natural-language processing (e.g., spoken language understanding)
6. market segmentation (e.g.: predict if customer will respond to promotion)
7. bioinformatics (e.g., classify proteins according to their function)

2.1.1 Machine Learning Algorithms

1. support-vector machines

2. neural networks

3. random forests

4. naive bayes

5. nearest neighbor algorithms

2.1.2 Support Vector Machines:

Support Vector Machine (SVM)[1] is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well.

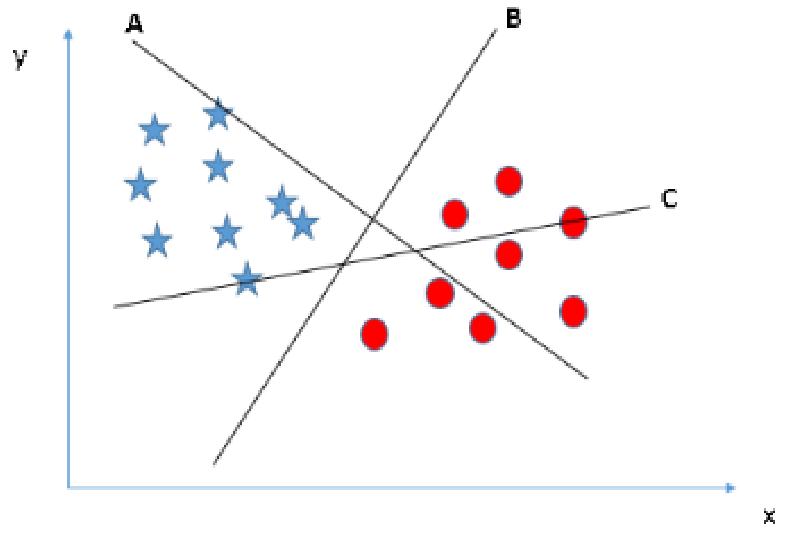


Figure 2: Support Vector Machines

In the figure 2, "Select the hyper-plane which segregates the two classes better". In this scenario, hyper-plane "B" has excellently performed this job.

In the figure 3 we have three hyper-planes (A, B and C) and all are segregating the classes well. Here, maximizing the distances between nearest data point

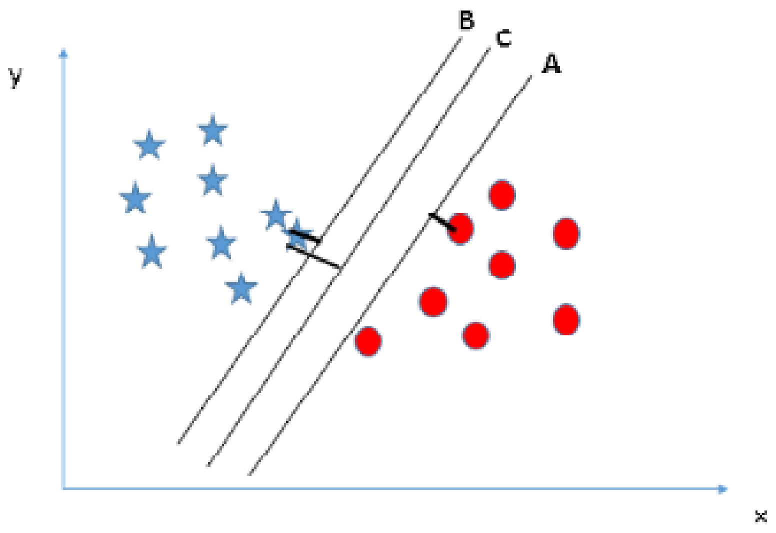


Figure 3: Support Vector Machines

(either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as Margin. We can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C.

SVM can find hyperplane in figure 4. It solves this problem by introducing additional feature. Here, we will add a new feature z=x+y. Now, let's plot the data points on axis x and z:

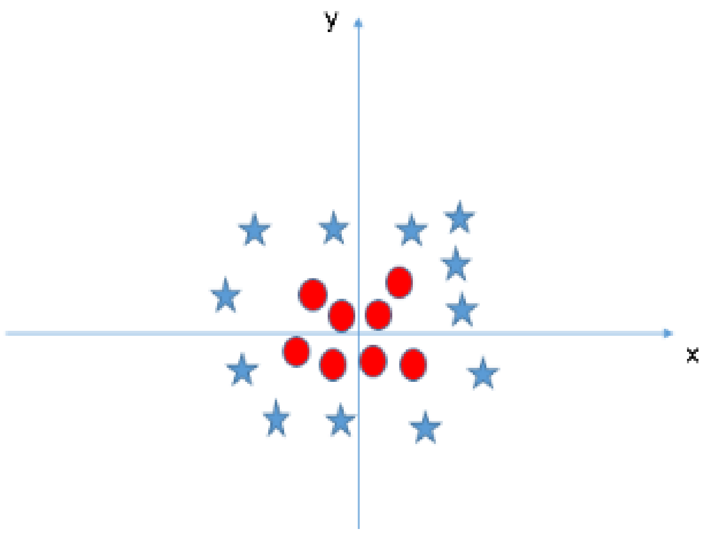


Figure 4: Support Vector Machines

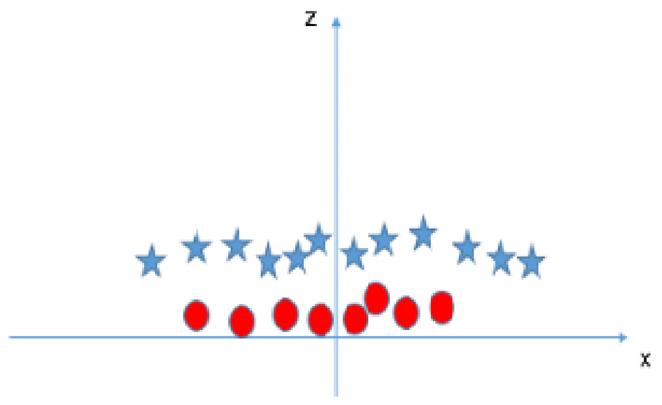


Figure 5: Support Vector Machines

In SVM, it is easy to have a linear hyper-plane between these two classes. But, another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, SVM has a technique called the kernel trick. These are functions which takes low dimensional input space and transform it to a higher dimensional space i.e. it converts not separable problem to separable problem, these functions are called kernels. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then find out the process to separate the data based on the labels or outputs you've defined.

Pros and Cons associated with SVM:

Pros:

1. It works really well with clear margin of separation
2. It is effective in high dimensional spaces
3. It is effective in cases where number of dimensions is greater than the number of samples
4. It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient

Cons:

1. It doesn't perform well, when we have large data set because the required training time is higher

2. It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping
3. SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation.

2.1.3 Neural Networks

Neural networks[2] are the building blocks of today's technological breakthrough in the field of Deep Learning. A neural network can be seen as simple processing unit that is massively parallel, capable to store knowledge and apply this knowledge to make predictions.

Single Layer Perceptron (SLP):

The simplest type of perceptron has a single layer of weights connecting the inputs and output. In this way, it can be considered the simplest kind of feed-forward network. In a feed forward network, the information always moves in one direction; it never goes backwards.

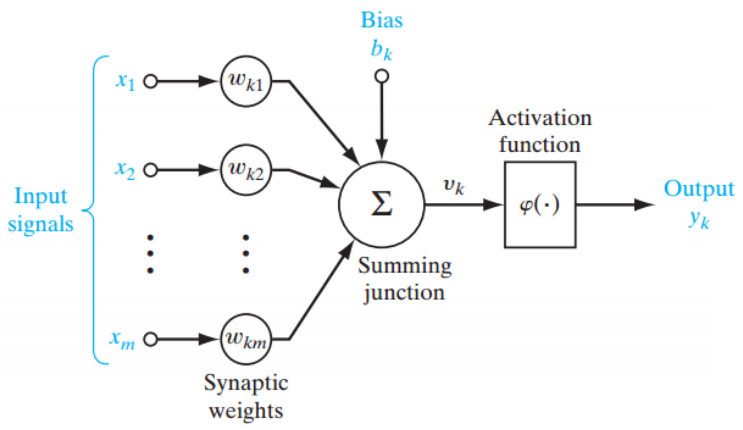


Figure 6: Single Layer Perceptron

Multilayer Perceptron (MLP):

Moving onwards, multi-layer perceptron, also known as feed-forward neural networks, consists of a sequence of layers each fully connected to the next one.

A multilayer perceptron (MLP) has one or more hidden layers along with the input and output layers, each layer contains several neurons that interconnect with each other by weight links. The number of neurons in the input layer will be the number of attributes in the dataset, neurons in the output layer will be the number of classes given in the dataset.

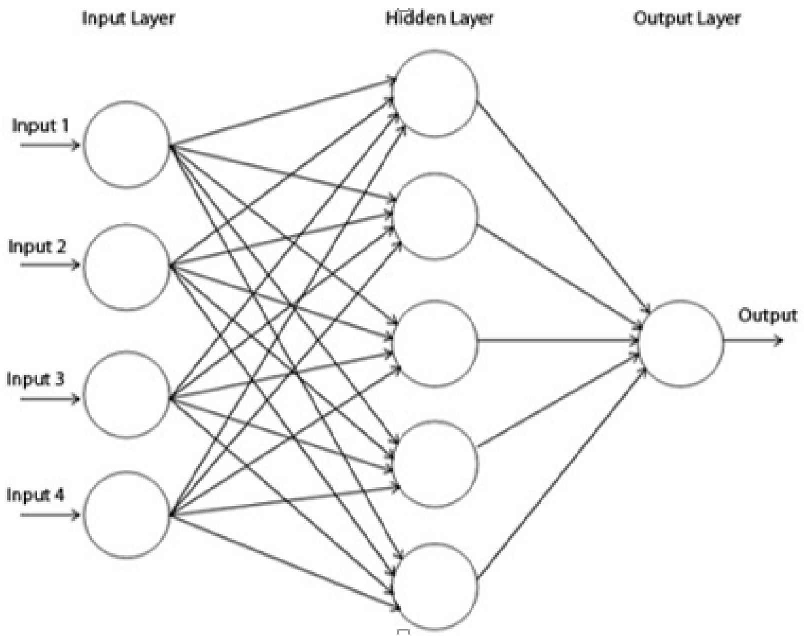


Figure 7: Multi Layer Perceptron

Activation Function:

1. Threshold Function
2. Sigmoid Function
3. Hyperbolic Tangent Function
4. Rectified Linear Activation function (ReLU)

Backpropagation Algorithm:

The back-propagation algorithm can be used to train feed forward neural networks or multilayer perceptrons. It is a method to minimize the cost function by changing weights and biases in the network. To learn and make better predictions, a number of epochs (training cycles) are executed where the error determined by the cost function is backward propagated by gradient descent until a sufficiently small error is achieved.

Advantages of NN:

1. ability to generalise and to respond to unexpected inputs/patterns
2. massive parallel processing with simple steps

2.1.4 Random Forests

Random Forests[3] grows many classification trees. To classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification, and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).
Each tree is grown as follows:

1. If the number of cases in the training set is N, sample N cases at random - but with replacement, from the original data. This sample will be the training set for growing the tree.
2. If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.
3. Each tree is grown to the largest extent possible. There is no pruning.

Features of Random Forests:

1. It is unexcelled in accuracy among current algorithms.
2. It runs efficiently on large data bases.
3. It can handle thousands of input variables without variable deletion.
4. It gives estimates of what variables are important in the classification.
5. It generates an internal unbiased estimate of the generalization error as the forest building progresses.
6. It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing.
7. It has methods for balancing error in class population unbalanced data sets.
8. Generated forests can be saved for future use on other data.
9. Prototypes are computed that give information about the relation between the variables and the classification.
10. It computes proximities between pairs of cases that can be used in clustering, locating outliers, or (by scaling) give interesting views of the data.
11. The capabilities of the above can be extended to unlabeled data, leading to unsupervised clustering, data views and outlier detection.
12. It offers an experimental method for detecting variable interactions.

2.2 Information Extraction[6]

It simply means getting simple structured information out of text.

Information extraction (IE) systems:

1. Find and understand limited relevant parts of texts
2. Gather information from many pieces of text

3. Produce a structured representation of relevant information:

   a) relations (in the database sense)

   b) a knowledge base

4. Goals:

   a) Organize information so that it is useful to people

   b) Put information in a semantically precise form that allows further inferences to be made by computer algorithms

E.g., Gathering earnings, profits, board members, headquarters, etc. from company reports:
The headquarters of BHP Billiton Limited, and the global headquarters of the combined BHP Billiton Group, are located in Melbourne, Australia.

headquarters("BHP Biliton Limited", "Melbourne, Australia")

2.3 Named Entity Recognition (NER)[5]

A very important sub-task: find and classify names in text.

The uses:

1. Named entities can be indexed, linked off, etc.
2. Sentiment can be attributed to companies or products
3. A lot of IE relations are associations between named entities
4. For question answering, answers are often named entities.

Concretely:

1. Many web pages tag various entities, with links to bio or topic pages, etc.

2. Reuters' OpenCalais, Evri, AlchemyAPI, Yahoo's Term Extraction, ... Apple/Google/Microsoft/... smart recognizers for document content

Three standard approaches to NER (and IE):

1. Hand-written regular expressions
2. Using classifiers
   a) Generative: Naïve Bayes
   b) Discriminative: Maxent models
3. Sequence models i.e. HMMs
4. Part-of-speech (POS) tagging
   Mark each word as a noun, verb, preposition, etc.
5. Syntactic parsing
   Identify phrases: NP, VP, PP
6. Semantic word categories (e.g. from WordNet)
   KILL: kill, murder, assassinate, strangle, suffocate

3 Software Requirements Specification

3.1 Introduction

Mathematical word problems (MWP) test critical aspects of reading comprehension in conjunction with generating a solution that agrees with the "story" in the problem. We design and construct an MWP solver in a systematic manner, as a step towards enabling comprehension in mathematics and teaching problem solving for children in the elementary grades. We build a multistage software prototype that predicts the problem type, identifies the function of sentences in each problem, and extracts the necessary information from the question to generate the corresponding mathematical equation.

3.1.1 Project Scope

Since there could be variety of MWP and it will be a tough job to define a generic solver. So the system focuses on solving three types of problems:

1. Join and Seperate Problems
2. Part-Part-Whole problems
3. Comparison Problems

3.1.2 User Classes and Characteristics

Two user classes exists for our project:

1. Young Learners-For intelligent tutoring system
2. Cognitive Scientists-To understand the cognitive aspects of problem solving in children

3.1.3 Operating Environment

For Automatic Problem Solver, operating environments are:

1. Operating System: Linux and Unix
2. Python and its NLP libraries
3. Training dataset is present in JSON file

3.1.4 Design and Implementation Constraints

Following are the Design and Implementation Constraints:

1. How well the classifier is predicting problem type?
2. How well a sentence function is predicted?
3. Whether equation generated is correct or not?
4. When the problem relates to time, money and distance, it needs quantity conversions before the arithmetic calculations

3.1.5 Assumptions and Dependencies

We are assuming following to design automatic problem solver:

1. Questions do not have the complex sentence structure.
2. Only additions and subtractions type problems are solved by solver
3. Classifier is good enough to predict problem type

3.2 System Features

There could be lots of features for our system. Some of them are listed below:

1. Solver to MWP
2. Step by step representation of solution(optional)
3. Web interface to solver(optional)

3.2.1 Solver to MWP(Functional Requirements)

1. A large number of training dataset in JSON is required to train the classification model.
2. A well-trained information extractor

3.2.2 Step by step representation of solution(Functional Requirements)

1. A module to capture the intermediate processing steps of a problem and finally display them.

3.2.3 Web interface to solver(Functional Requirements)

1. Deploying the solver to web using Web Technologies.

3.3 External Interface Requirements

3.3.1 User Interfaces

It is the secondary part of the project as main emphasis is being given to design a solver using CLI.

3.3.2 Hardware Interfaces

1. CPU with 2GHz processing power
2. GPU(NVIDIA) for parallel processing(optional)
3. RAM 2GB

3.3.3 Software Interfaces

1. Operating System: Linux(Ubuntu)
2. Programming Language: Python3
3. Libraries Used: scikit, NLTK3, TensorFlow

3.3.4 Communication Interfaces

Web interface is supported across all web browsers.

3.4 Non-functional Requirements

3.4.1 Performance Requirements

The Solver should solve the problem correctly. For this we weed a well-trained Classifier.

3.4.2 Safety Requirements

Classifier should be saved somewhere else(back up) for future use of similar kind of work and also for extending this work.

3.4.3 Security Requirements

No Security Threats as system does not contain any sensitive data of users.

3.4.4 Software Quality Attributes

1. AVAILABILITY: System should be available to as many customers who are using tutoring system.
2. CORRECTNESS: System should solve the MWP correctly.

3. MAINTAINABILITY: The administrators should be able to re-train the sytem if needed.
4. USABILITY: System should solve almost all problems of given domains.

3.5 Analysis Models

Various diagrams of the system like Data Flow Diagrams, Class Diagrams, State-transition Diagrams are useful to analyze the system functionaly and implementation details.

3.5.1 Data Flow Diagrams

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated.

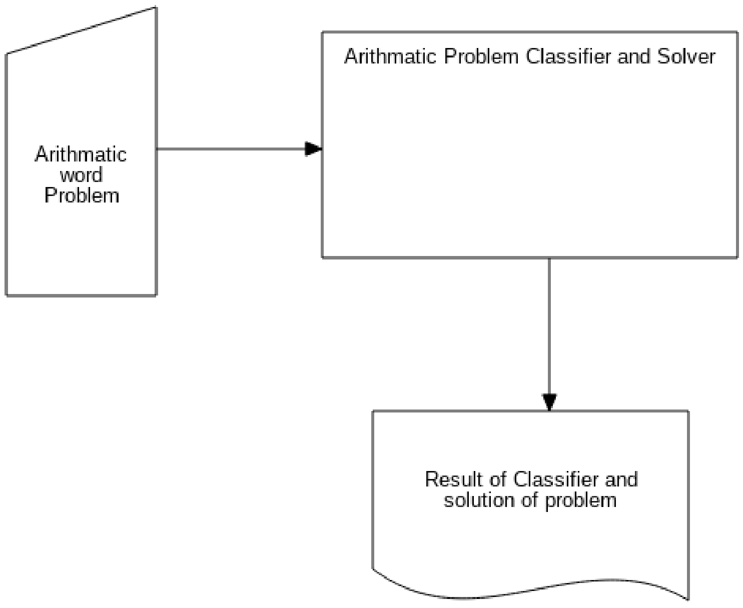

Arithmatic word Problem
Arithmatic Problem Classifier and Solver
Result of Classifier and solution of problem

Figure 8: Level-0 DFD

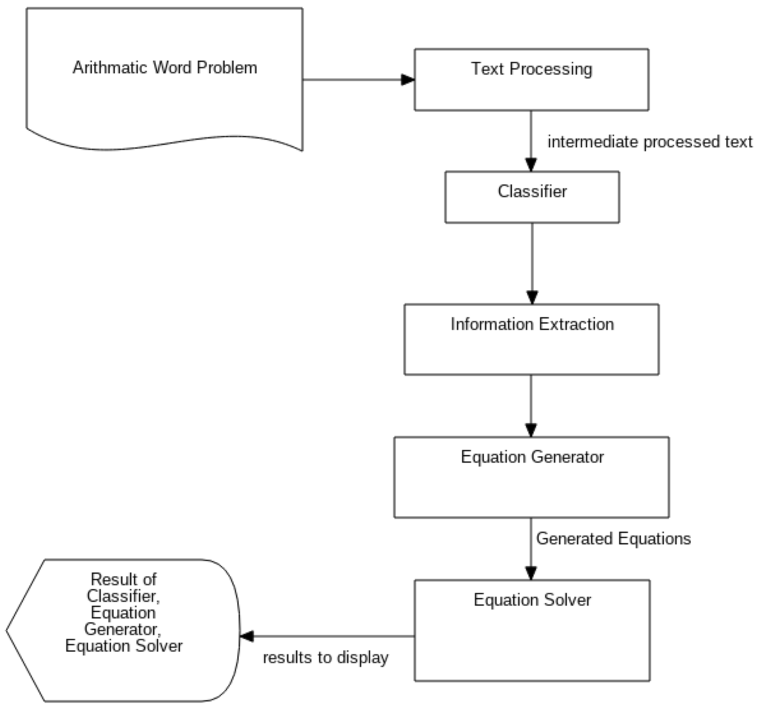


Figure 9: Level-1 DFD

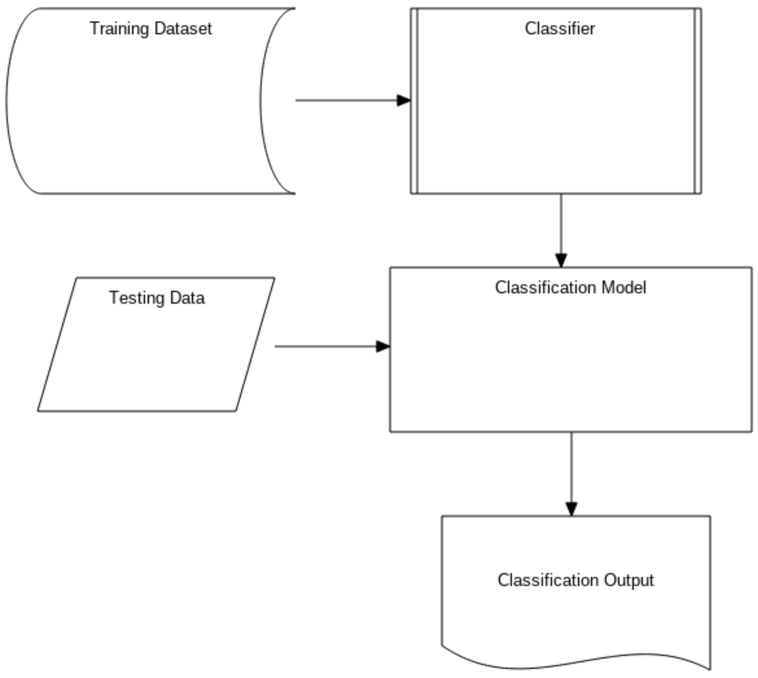


Figure 10: Level-2 DFD

3.5.2 Class Diagrams

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

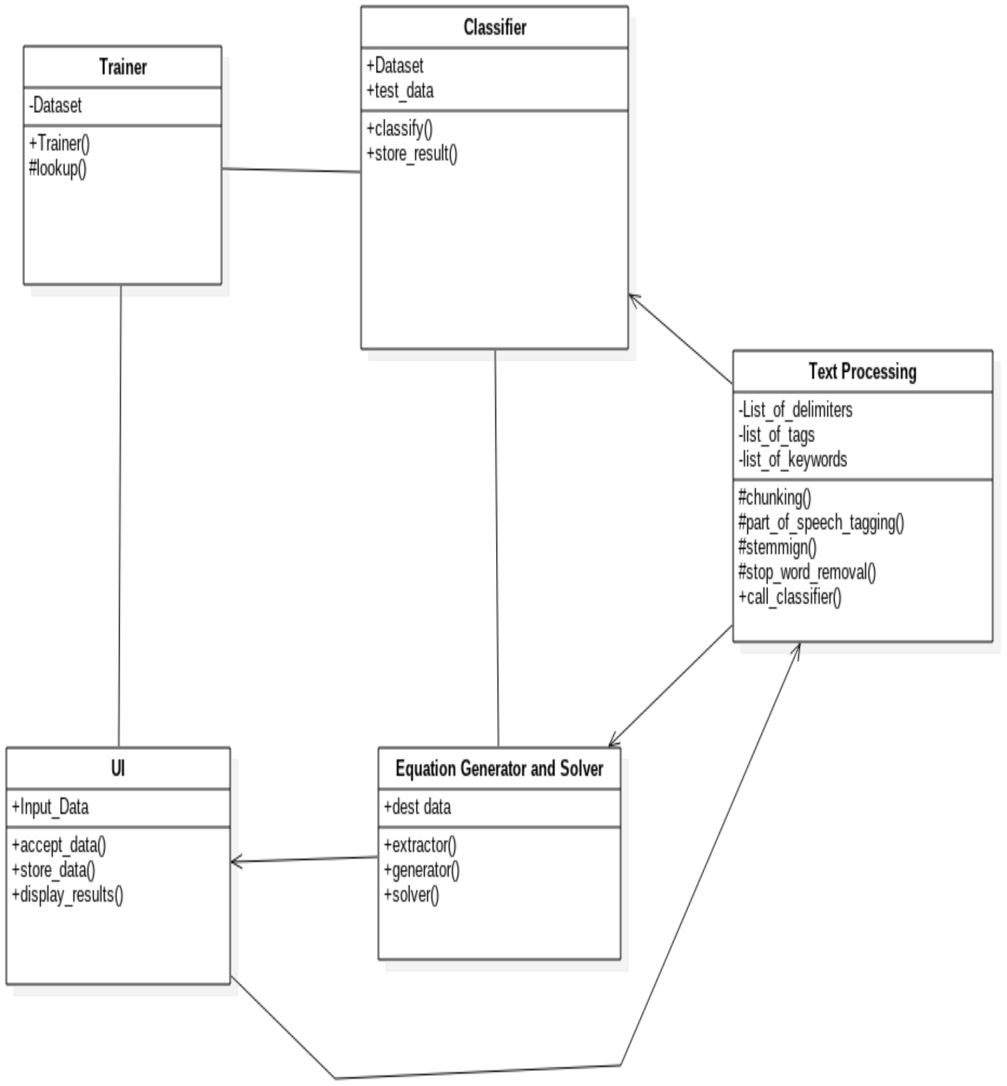


Figure 11: Class Diagram

3.5.3 State-transition Diagrams

A state diagram is a type of diagram used in computer science and related fields to describe the behavior of systems. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.

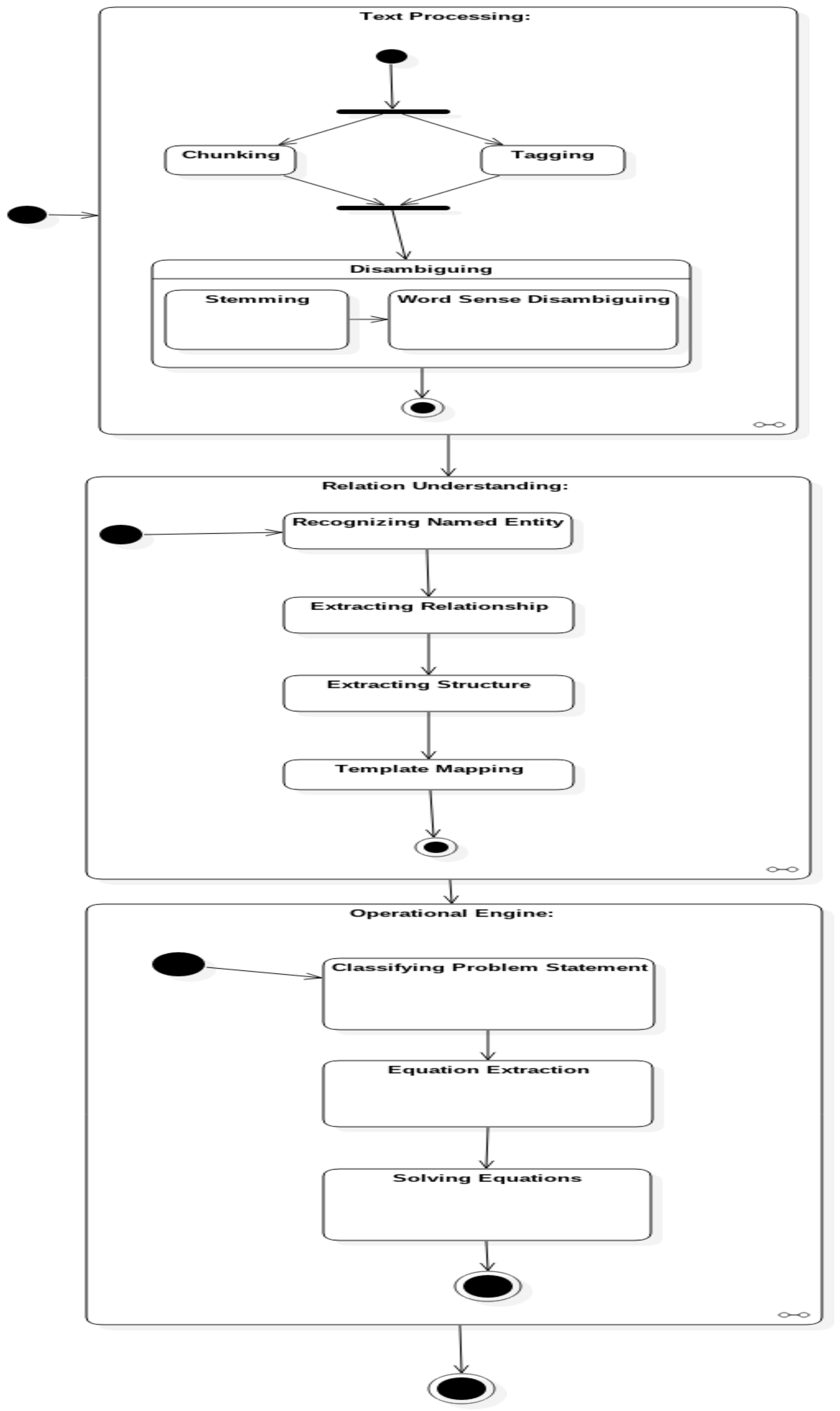


Figure 12: State Transition Diagram

3.6 System Implementation Plan

Table 1: System Implementation Plan

Month	**Schedule**	**Project Task**
July	1st Week	Idea about Project Selection/Project Topic Selection
	3rd Week	Submission of Project Synopsis/Abstract
	5th Week	Guide Allocation & Literature Survey
August	1st Week	To Display List of Project Groups with Guide Names
	2nd Week	First presentation with Guide about idea of projects (Feasibility study)
	3rd & 4th Week	Requirement Analysis (SRS Document) Preparation & Submission
September	1st Week	Design of Project-Low level, High Level, Data Structure, Database tables & Algorithms
	3rd Week	Presentation –II with design
	4th Week	Preparation of preliminary report
October	1st Week	Submission-Prelim Report of the Project
November	2nd Week	University Exam on Preliminary Report
January	1st Week	Coding
	2nd Week	At lest 2 module should finish (30%) Total Work
February	1st Week	First demonstration on project work expected (60%) of total work
March	1st Week	Test Plan , Design & Installation
	3rd Week	Final Project Demonstration
	4th Week	Preparation of project report , Preparation of Installable Project & Manual
April	1st Week	Submission of Report (Final Submission)
May	2n Week	Final University Examination

4 System Design

4.1 System Architecture

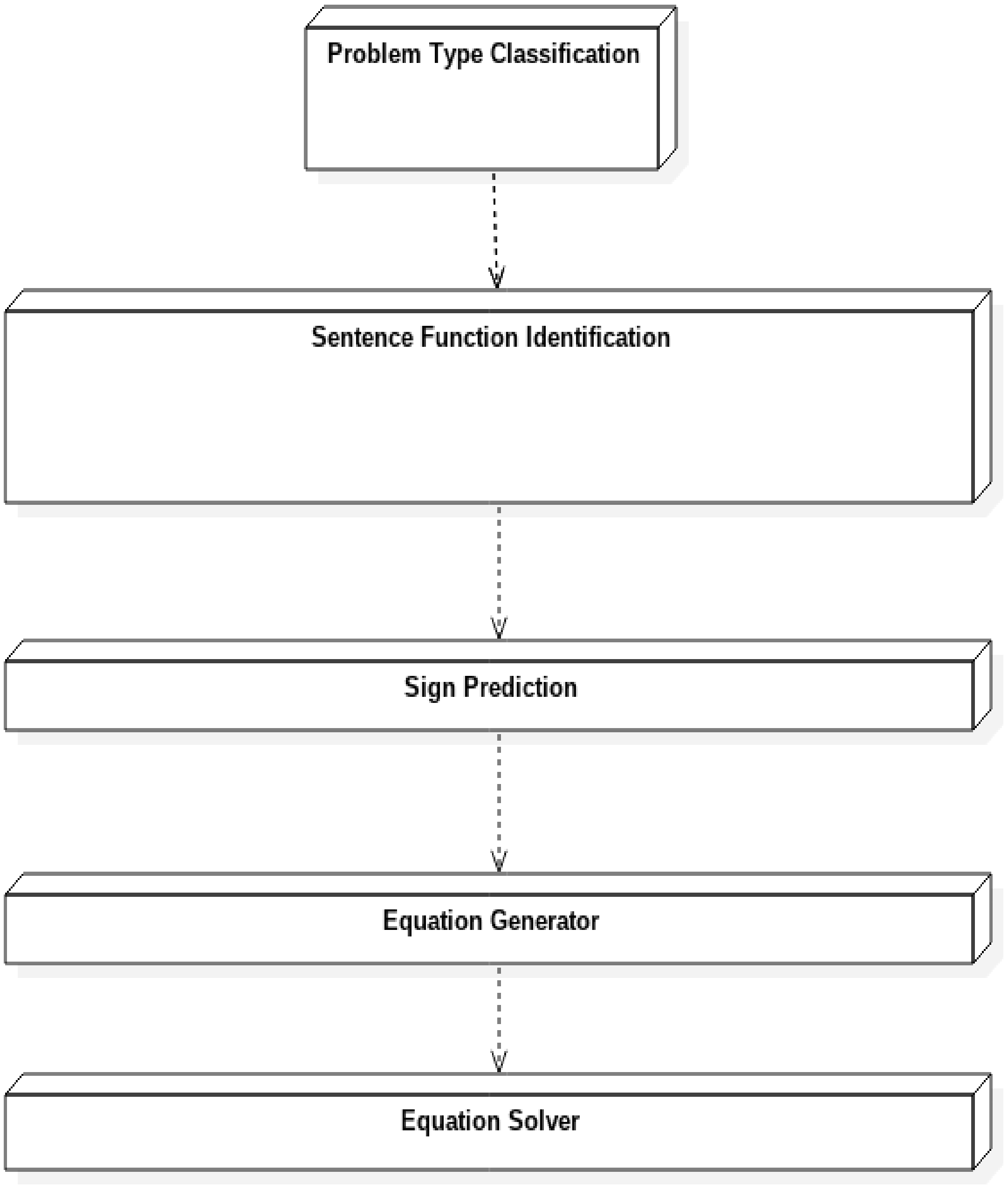


Figure 13: System Architecture

4.2 Algorithm

1. Accept a algebric word problem
2. Classify it into Join and Seperate problem or Part-Part-Whole problem or Comparison problem
3. Identify function of each sentence in the problem
4. Generate equations(with the help of Information Extraction)
5. Solve Equations
6. Display Solutions

4.3 UML Diagrams

4.3.1 Activity Diagram

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system.

4.3.2 Use Case Diagram

Use case diagram is dynamic in nature and there should be some internal or external factors for making the interaction.These internal and external agents are known as actors. So use case diagrams are consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application.

4.3.3 Sequence Diagram

A Sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart.

4.3.4 Package Diagram

Package diagram is UML structure diagram which shows packages and dependencies between the packages.

4.3.5 Component Diagram

Component diagrams are different in terms of nature and behaviour. Component diagrams are used to model physical aspects of a system. Now the question is what are these physical aspects? Physical aspects are the elements like executables, libraries, files, documents etc which resides in a node.

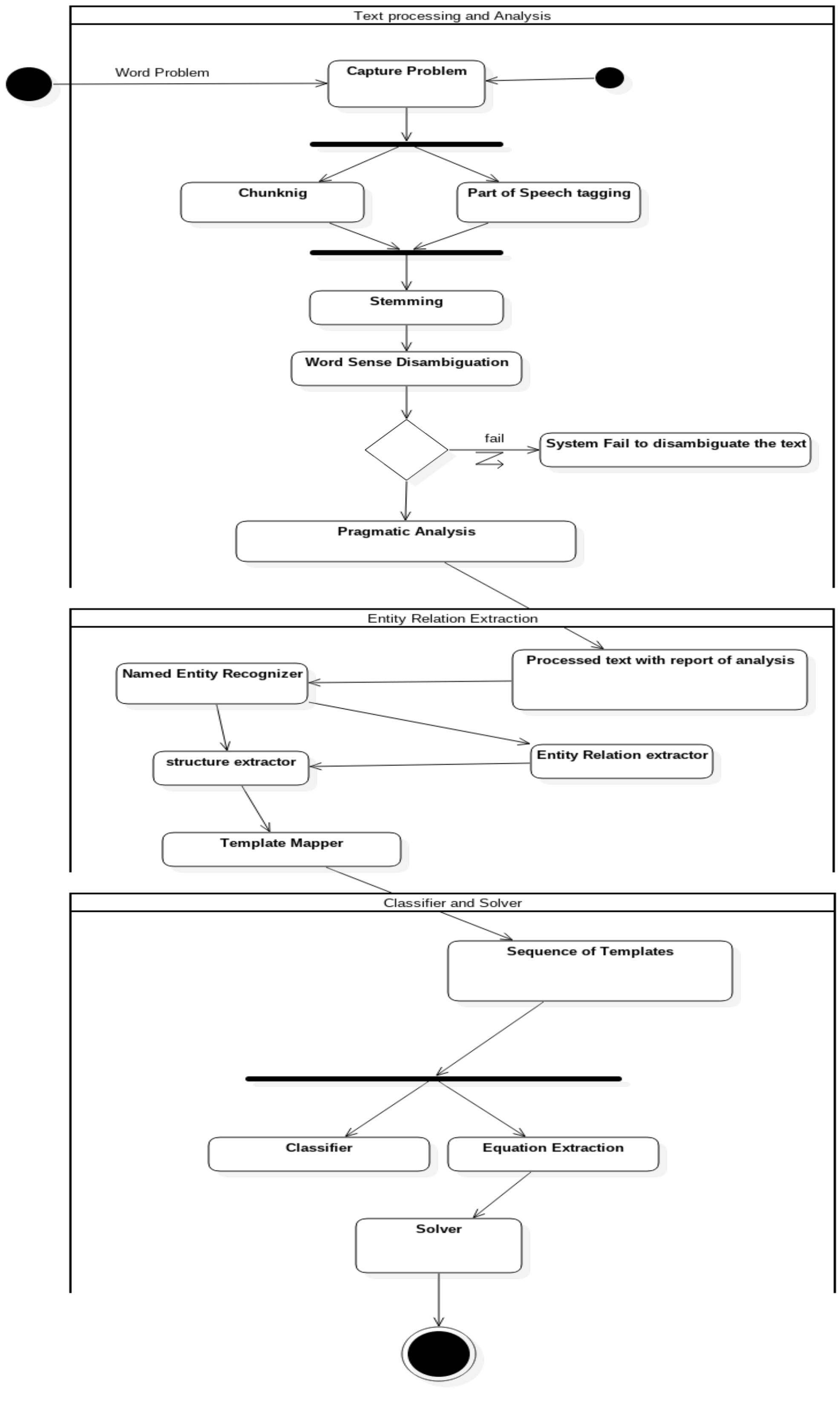


Figure 14: Activity Diagram

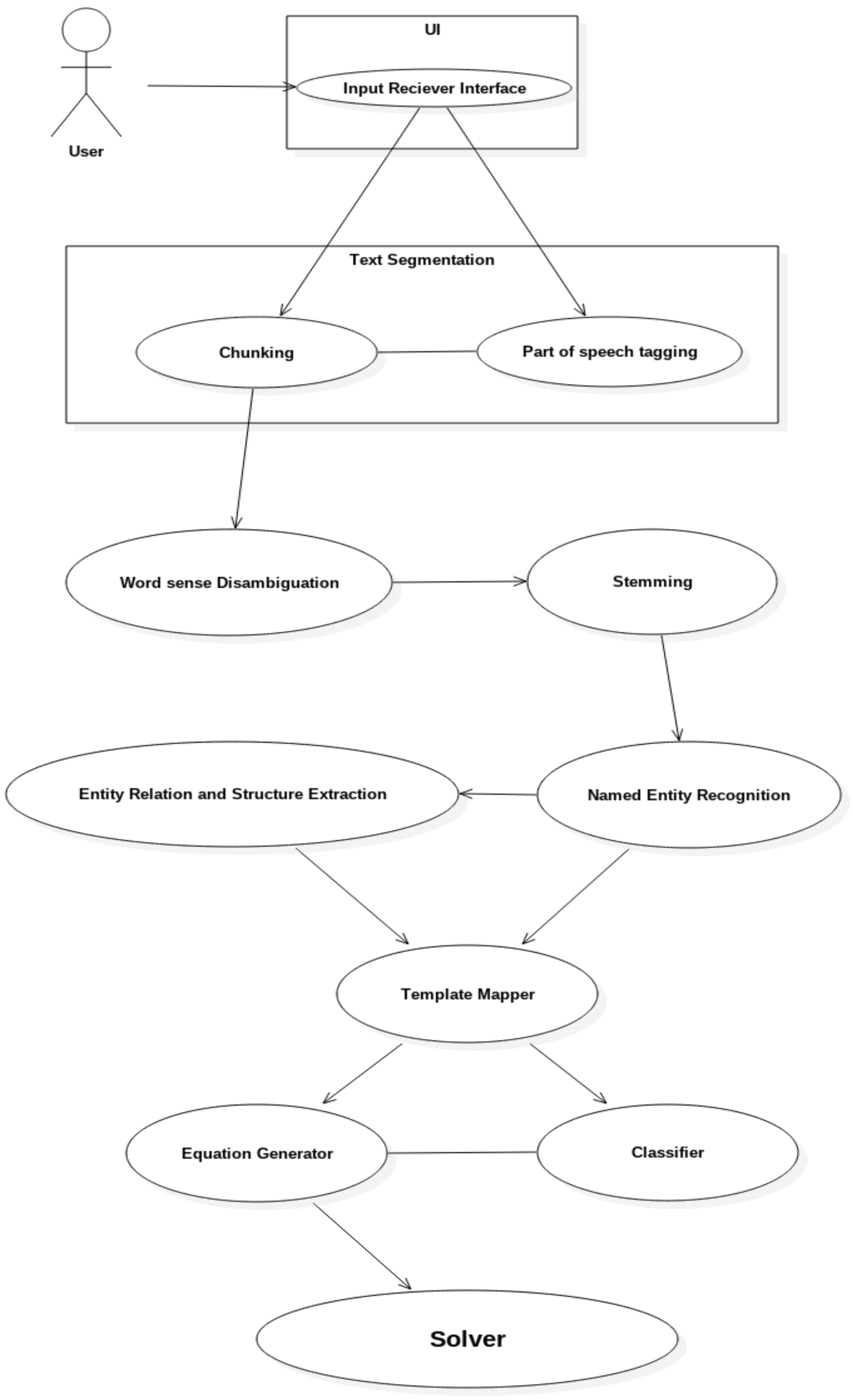


Figure 15: Use Case Diagram

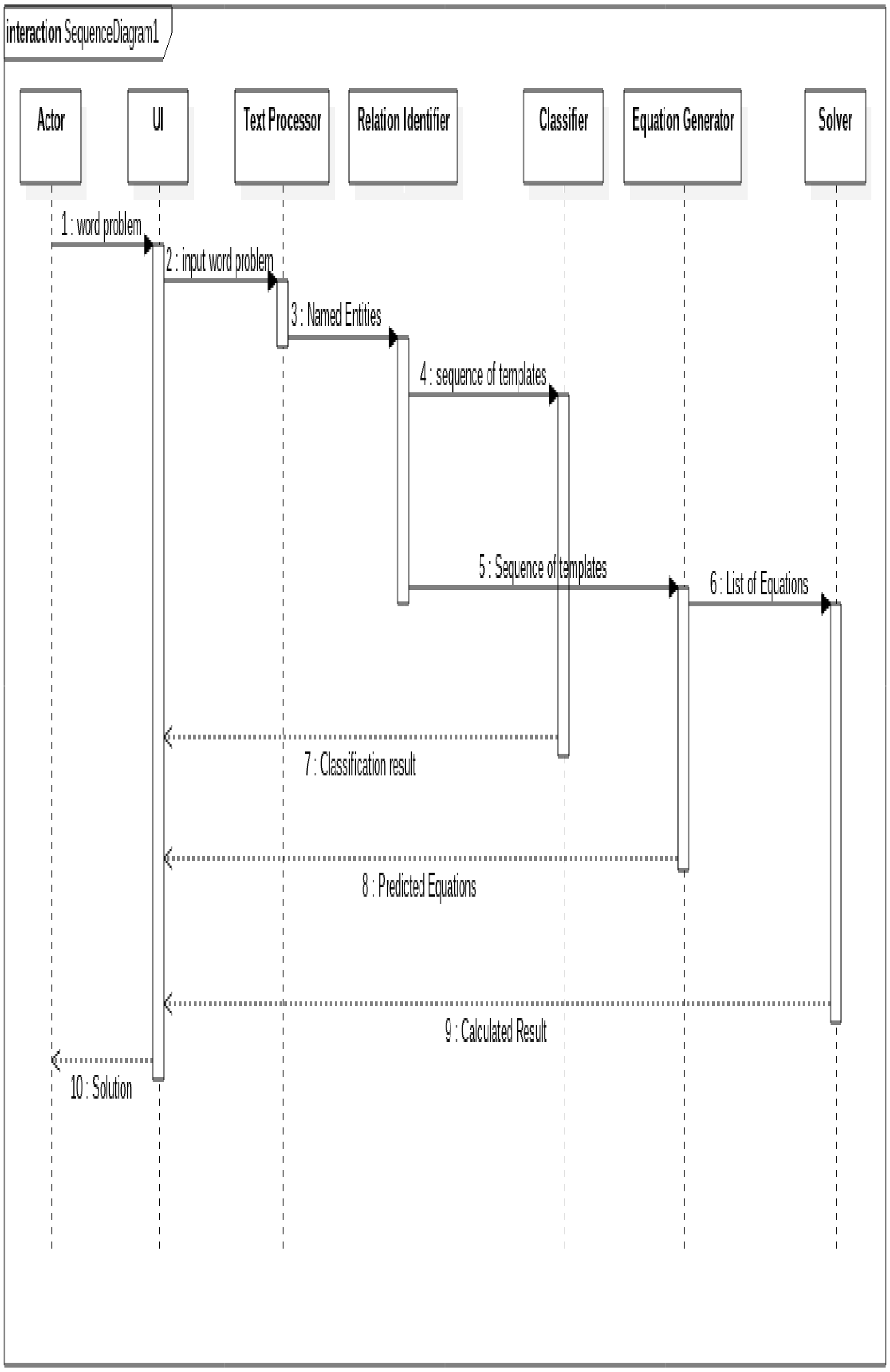


Figure 16: Sequence Diagram

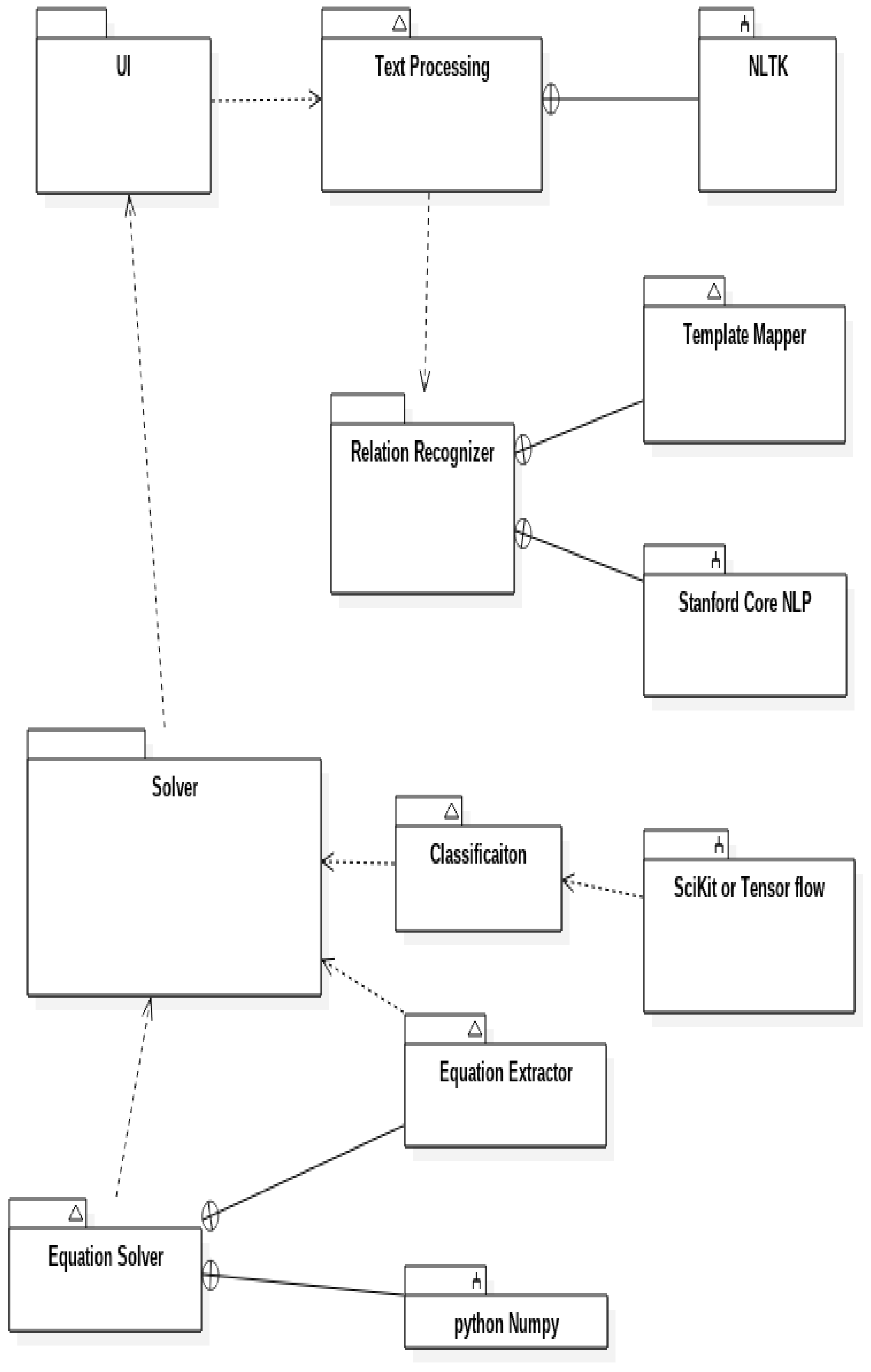


Figure 17: Package Diagram

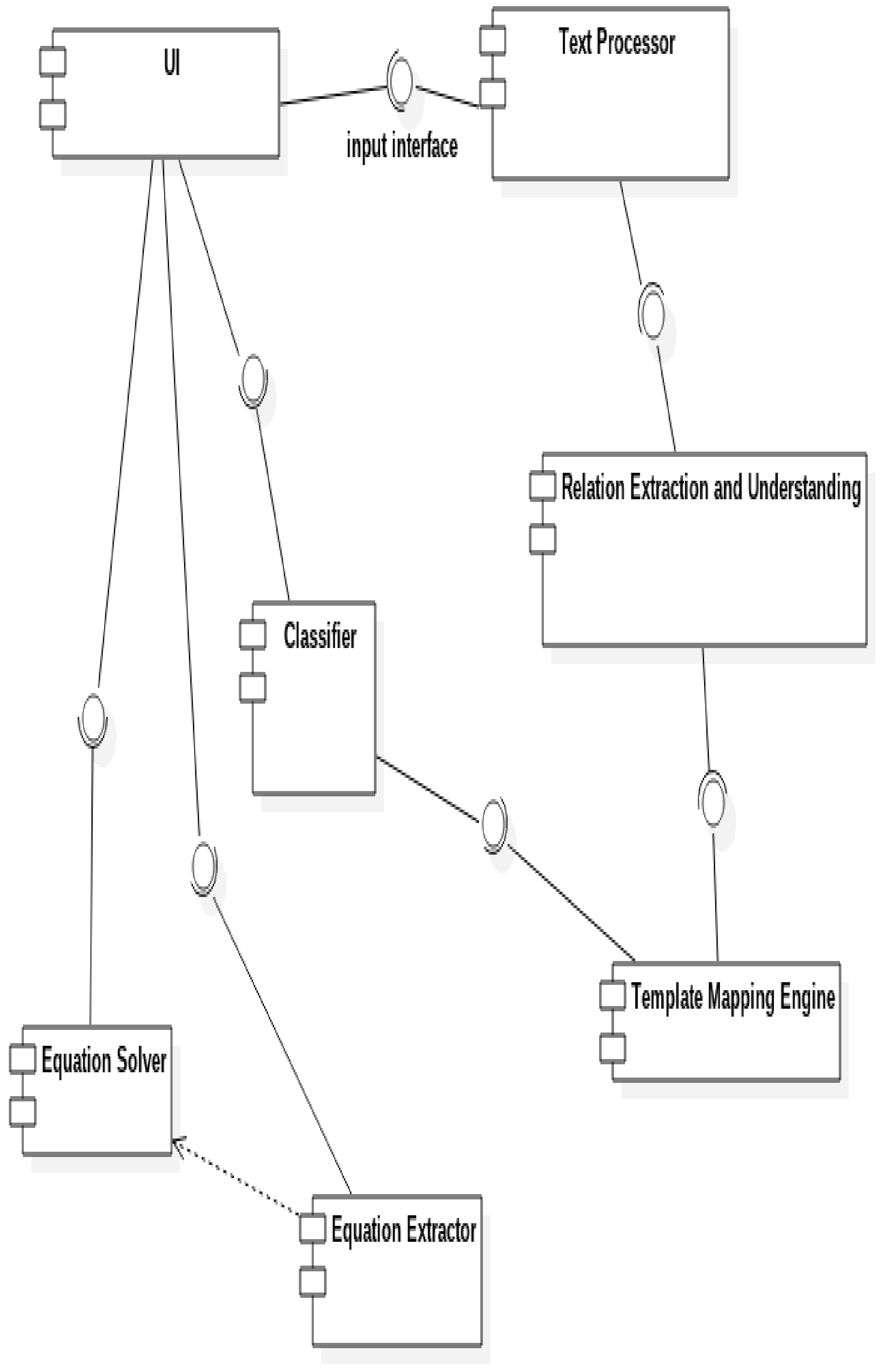


Figure 18: Component Diagram

5 Technical Specification

5.1 Advantages

1. Assists students to solve MWP
2. Assists cognitive scientists to understand problem solving in children
3. More similar systems could be developed in future using analogy like automatic algorithm and data structure prediction for computer problems.
4. It covers in detail the concept of Classification and Information Extraction which is useful in many NLP projects.

5.2 Disadvantages

1. Chances of wrong classification of problem type for a MWP
2. Chances of wrong equation generation
3. Chances of wrong sign prediction in Join and Seperate problems

5.3 Applications

1. Intelligent Tutoring Systems
2. To study how children solve elementary MWP by Cognitive Scientists
3. Web based assistant for MWP

6 Appendix A: Assignments

7 Appendix B: Glossary

List of Acronyms:

1. MWP - Mathematical Word Problem
2. NN - Neural Network
3. SVM - Support Vector Machine
4. NER - Named Entity Recognition
5. IE - Information Extraction
6. UML - Unified Modelling Language

8 References

[1] Jetlin.C.P, Mercy.W and Dr. P.S.K Patra, "Classification of Questions in Microblogging Environment Using Support Vector Machine" in International Journal of Research in Computer and Communication Technology, Vol 3, Issue 4, April- 2014.

[2] Peter G. Zhang, "Neural Networks for Classification: A Survey" in in IEEE Transactions on Systems Man and Cybernetics Part C (Applications and Reviews), December 2000

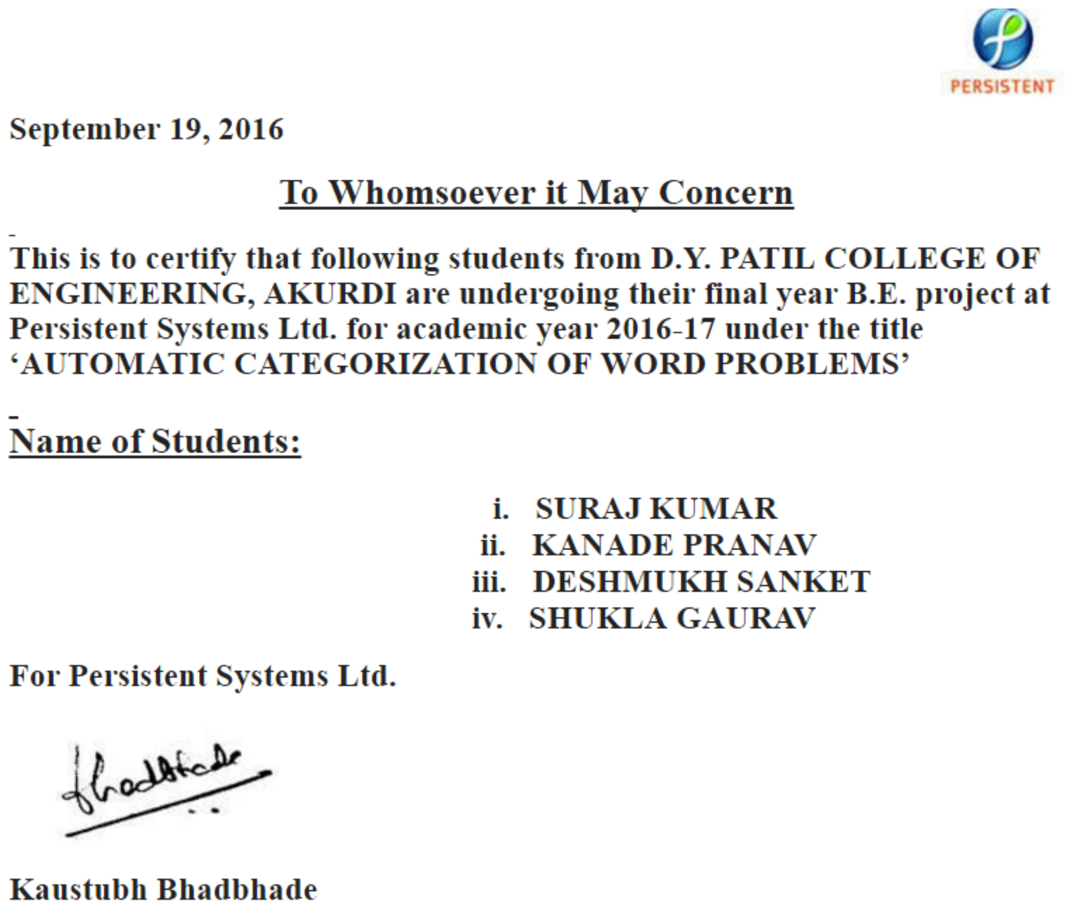
[3] Andy Liaw and Matthew Wiener, "Classification and Regression by randomForest" Vol. 2/3, December 2002

[4] Yiming Yang and Xin Liu, "A re-examination of text categorization methods" in Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, pages 42–49. ACM, 1999.

[5] D. Nadeau and S. Sekine, "A Survey of Named Entity Recognition and Classification," Linguisticae Investigationes, vol.30, 2007, pp. 3–26.

[6] Ronan Collobert, JasonWeston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, Pavel Kuksa, "Natural Language Processing (Almost) from Scratch" in Journal of Machine Learning Research 12 (2011) 2493-2537 in 2011.

9 Sponsorship letter



PERSISTENT

September 19, 2016

To Whomsoever it May Concern

This is to certify that following students from D.Y. PATIL COLLEGE OF ENGINEERING, AKURDI are undergoing their final year B.E. project at Persistent Systems Ltd. for academic year 2016-17 under the title 'AUTOMATIC CATEGORIZATION OF WORD PROBLEMS'

Name of Students:

i. SURAJ KUMAR
ii. KANADE PRANAV
iii. DESHMUKH SANKET
iv. SHUKLA GAURAV

For Persistent Systems Ltd.

[illegible]

Kaustubh Bhadbhade

Figure 19: Sponsorship letter