TECHNISCHE UNIVERSITÄT
CHEMNITZ

Multi Criteria Mapping Based on SVM and Clustering Methods

Master Thesis

for
the fulfilment of the academics degree

M.Sc. in Automotive Software Engineering

Faculty of Computer Science

Professorship of Computer Engineering

August 2015

Submitted by: Abhishek Diddikadi, Matr. Nr. 335001

Supervisors: Prof. Dr. Wolfram Hardt
Dipl.-Inf. Daniel Reißner

Abhishek Diddikadi (abhishek.diddikdadi@informatik.tu-chemnitz.de)
Multi Criteria Mapping Based on SVM and Clustering Methods
Master Thesis, Technische Universität Chemnitz, August 2015.

Acknowledgment

This Master Thesis would not exist without the extensive support and encouragement of many people. I express my intense gratitude and whole hearted thanks to supervisor **Dip.-Inf. Daniel Reißner** for his useful ideas, suggestions, time and contribution, without him this research wouldn't have started and completed on time. Special thanks to a person very much near and dear to student **Prof. Dr. Wolfram Hardt** for his guidance, immense support from the day I stared my master course. I thank you for providing me an opportunity to take up this research work under your professorship and for your supervision till it came to an end.

In addition to these and all other people at the Technische Universität Chemnitz I might not forget my family who tolerated my absence in several activities in family life. Thanks a lot to my soul mate for your tremendous patience and keeping me focused towards my work. I would appreciate my friends for keeping me more energetic and there support remains with me forever.

Thank you.

Table of Contents

Page

List of Figures & Tables

Chapter 1

Introduction

1.1 Motivation

There was an increase in the number of applications for master's program with the increase in time. For processing all these applications manually, high volume of time and workforce is required. This can be reduced if automation is used in the process. But prior to that, an analysis of the complete steps involved in processing and were exactly the automation has to be used to decrease the time and work forces must be made. The applicant process actually involved several steps. First, the applicant sends the complete documentation to uni-assist; from there the applications are received by the student assistance team at university; and are then sent to the individual departments. At the individual departments, the individual applications will be processed by conducting a thorough study on if the applicant has required skills in his previous studies and if they are fit for ASE study program. With this thesis project a single web tool can be developed that can process the application which is much reliable in the decision making process of application.

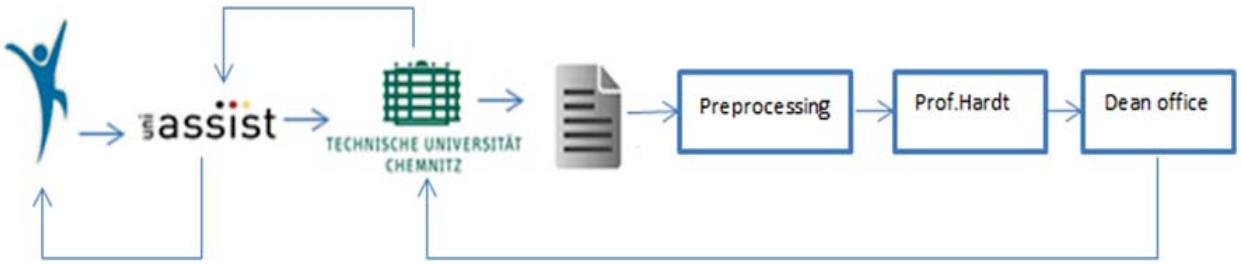


Fig 1.1: Complete application process

Here comes the manual processing of the application. In this process, we will receive a pdf document from the uni-assist containing each and every document submitted by the applicant. From these documents, the data about applicant personal details, previous educational details and also courses that can be matched to the ASE program for further studies will be manually entered in to the database by the auditors. There exist other parameters like work experience, education or course certifications and German language skills, which fetch some bonus points. These points are combined in the final state. If applicant manages to cross particular limit points, then an admission in ASE is given to the applicant. There are some web applications for processing the applications in this way. For any further examination, all the data received from the student and uni-assist is stored. As the number of applications to be processed increase, the process needs much time and number of auditors.

1.2 Problem description

The issue faced in manual processing of application is: Processing time of each application, manual data entry for further classification, manual analysis of score cards, less number of auditors, the less accuracy rate in classification. The main idea behind this thesis work would be to eradicate all this issue by introducing a tool that can automate pre-processing stage in this application process.

To automate the process I had a research work on many automation tools, text conversion tools and others as explained in state of the art. Finally come up with an idea to use OCR for text conversion and implement SVM and Cluster based approaches while classifying the data. Compare both the results to get best classifier. With this approach we can automate pre-processing stage of the application process and can see much time, save of repeated tasks auditors involved in this process. Furhter more semi automatic preselects and processing order organisation cross checks processing steps with auditors intention. Other advantages of this process will be easy data management, easy processing of the applications that are often repeated from the same university.

1.3 Project Information

This thesis is done in co-operation with professorship in Computer Engineering Department at Technical University of Chemnitz. The aim of this project is to develop a tool that can automatize the student application process. To implement this we used SVM and clustering based approaches. We analyze results by comparing both approaches with their results on how well they can identify ASE related courses. The database should be designed in such a way that it can be used for further developments with new approaches for artificial intelligence. This tool can be further extended to other study courses in the university.

1.4 Outline of Thesis

There are many more ways to automate the application process like using some commercial software's that are used in big organizations to scan bills and forms, but this application is only for the static frames or formats. In our application, we are trying to automate the non-static frames as the study certificate we get are from different counties with different universities. Each and every university have there one format of certificates, so we try developing a very new application that can commonly work for all the frames or formats. As we observe many applicants are from same university which have a common format of the certificate, if we implement this type of tools, then we can analyze this sort of certificates in a simple way within very less time. To make this process more accurate we try implementing SVM and Clustering methods. With these methods we can accurately map courses in certificates to ASE study path if not to exclude list. A grade calculation is done for courses which are mapped to an ASE list by separating the data for both labs and courses in it. At the end, we try to award some points, which includes points from ASE related courses, work experience, specialization certificates and German language skills. Finally, these points are provided to the chair to select the applicant for master course ASE.

1.5 Document Layout

Complete document of thesis is divided into five chapters. Each and every chapter handles different topics. Chapter 1 introduces the topic. Chapter 2 gives the background knowledge and research we made on previously existing methods. It also briefs some detailed knowledge in character recognition and machine learning methods that exits. Chapter 3 gives information about the complete concept involved in this thesis work. In this chapter, we discuss more about the modules arranged to obtain final results. Working of each and every module and their results that can be used in further processing. Chapter 4 tells about different the comparison results of both SVM and cluster methods we implementions in this thesis work. Chapter 6 gives an overview of further implements and phases with developments that can be performed.

Chapter 2

State of the Art

This chapter discusses about the overview of the related background knowledge we obtained with research work on different topics we handled.

I OCR (Optical Character Recognition)

Optical Character Recognition, abbreviates as OCR, is the process where in automatic conversion of the alphanumeric characters in a document into computer readable format. Character conversion from one domain into another domain started in early 1913's. In the same year, a device called 'Optophone' was invented. This device converts a Braille script or symbols present on a paper into voice tones for fast reading so that we can hear the Braille code. Later on, the technology was improved and developed the OCR in the early 1950. OCR currently encompasses two distinct areas – pure optical character recognition by making use of the optical techniques like mirrors and lenses and digital character recognition by making use of scanners and computer algorithms. There is a tremendous change in the field, from an application which requires font-specific training, to the present generation's intelligent application which is capable of recognizing most fonts with a high degree of accuracy. Most of these intelligent applications use some forms of machine learning to adapt and learn to recognize new fonts. The Current OCR machine recognizes the computer documents with an accuracy of over 97%, and the clearly handwritten text on clear paper with 80-90%. Recognition of cursive text is an active area of research.

2.1 Construction of OCR system

The types of OCR systems are characterized by certain features. The types of OCR ate: Structural type, feature type, and neural network type. Some implementations employ State Vector Machine (SVM) along with the wavelets as the input data for the OCR process. So as to drive the wavelets, these methods involve much complex mathematical calculations. These in turn employ matrix multiplications and summations. One more important thing to be noted is, the SVM essentially requires time for taking different input-output pairs and calculating the relation between those input-output pairs. OCR types which use neural networks also use training techniques and some mathematical calculations, are also required for further processing. On the other hand, systems which do not use a training mechanism or memory based training or recognition are also available. A few of these systems uses fuzzy logic and histogram type area weight detection of the alphabet. But these have some disadvantages among which the prime is that these would be complex computation and the process is taking more time. Another type is feature recognition type OCRs, different features are extracted from the alphabet present in the input. This method is advantageous in terms of memory utilization and computations, as limited features are enough to recognize the character.

The figure of an OCR system is as shown below. Actually, it comprises several parts, but it is divided into four major parts namely:

- Pre-processing part
- Self-learning part
- Recognition part
- Post processing Part.

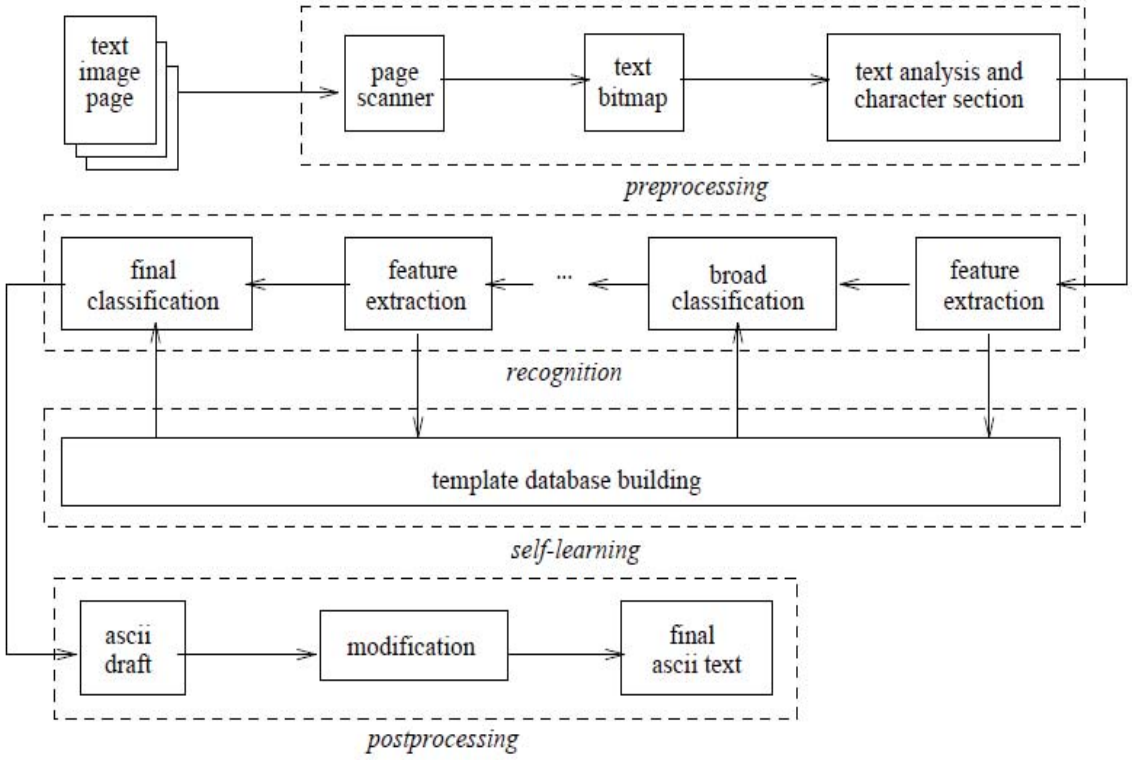


Fig 2.1: OCR system construction

2.1.1 Pre-processing part

The pre-processing part takes the input image obtained from the input sources like scanner or any other source and transfers the original text image into either a bitmap file or a binary matrix. Then the image undergoes a process called text analysis. In this process, the text image is sectioned into lines and characters. So as to remake the ASCII text file, the position of each character is recorded. Text analysis may encompass the following:

A. Extraction of character regions from an Image.

B. Segmentation of the image into text and background.

A) Extraction of Character Region from an Image

Extraction of character region from an image is accomplished by using the ancillary information known about the image to select the image properties which have sufficient variation of the text region and the background regions to be a basis of separating one from another. Once it is known, it is printed on a white or light background. We can make use of the detection line of text. The process can also be applied for segmenting the image into regions which contain words. A vertical hologram of the image is generated by a horizontal projection operator. The white space has a typical signature, which we use to segment the histogram and thereby the lines of text. Then we repeat the same procedure for each and every line of the text by making of the vertical projection operation for segmenting the words in each line of text [1].

Also, we can know the expected character size of the text, or other spatial organization or layout of the text. For identifying the areas of the image which have a character like spatial arrangements, we use either connect component analysis or mathematical morphology. We can select a mask so that morphology operations grow those characters for joining them to form word regions without bridging the space which is present between words or between lines. The next option available is by using connected component methods for identifying the image regions that have blobs of the right size. Utilizing the Local Fourier mask or texture masks, the areas of high spatial frequencies and hence segment textual regions can be identified.

B) Segmentation of the Image into Text and Background

While some OCR algorithms work with grayscale images, many converts their input into binary image formats during the early stages of processing. When an image region containing text, whether single word regions or whole slabs of text, is given, the goal of the stage is identifying the image pixels which belong to the text and the pixels which belong to the background. Thresholding of the gray-level image

is the most commonly used methods. We can choose the threshold value in many ways like using ancillary knowledge about the image, using statistical techniques to select a global threshold that 'best' classifies the image into two classes, and by using required measures calculated in the neighborhood of each pixel to determine the best local threshold. Practically, it is observed that the local adaptive thresholds produce the best segmentation results.

2.1.2 Self-learning Part

Every modern OCR tool has this special property of self-learning so as to enrich the knowledge whenever an unknown character string is obtained. It is recognized based on the database previously built in. This database contains most important feature related to the character which is already known. As more and more new characters are met, the recognition ability can be increased if the database is capable of self-expanding.

2.1.3. Recognition part

Of all the parts in the system, the Recognition part is the major one. It extracts input character features and compares them with the features that were recorded in the database used by the recognition. Whether the Characteristics would matched or nearly matched, the informative character will be ordered under those populations in which every last one of the characters needs these normal features. The last step in this stage is final classification. In this stage, the character stored in this block is written into result and the other characters fall under broad classification.

A) Feature Detection

Prior to feature detection, preliminary processing is often done. For any OCR, feature detection and classification are the heart. Since the advent of OCR several feature detection techniques have been used. First the whole character is found as a feature by template matching and then the sub features of

the character are sought. The algorithm found the boundary outlines, the character skeleton or medial axis, the Fourier or Wavelet coefficients of the spatial pattern, various moments both spatial and gray-level, and topological properties such as the number of holes in a pattern. All have been used as features for classification of the character regions.

While selecting character features to be detected, algorithm designers were pretty much aware of the importance of detecting the features that characterized the characters being read independently of actual font and size. On the other hand, the algorithm designers were attracted to those features which exhibit no variation with image variations, like translation, rotation, contrast and color. As a matter of fact each and every desired property of a feature can be found in any single measurement. Consequently, OCR algorithms often detect a range of features. The total of these features will be sufficient enough to identify each character irrespective of the specific instantiation used in the text. For detection of features, statistical correlation, boundary following, region thinning, mathematical morphology, and also different transforms like the Fourier transform, texture mask, spatial pattern operations, and topological algorithms are used. Prior to character segmentation, feature detection techniques are applied. Generally, these techniques attempt to find evidence for a specific character being present instead of the part features. Researches on finding new techniques for finding characters in text without prior character segmentation are increasing rapidly. The driving force behind the researches is the problems that were encountered in cursive scripts where segmentation without recognition may turn out to be a bigger problem than the recognition itself.

2.1.4. Post processing stage

The prime purpose of Post processing stage is to make necessary corrections, if any required. The ASCII text consists both recognized and rejected character and the character requires further editing and

modification. So the resulting character which is obtained after recognition stage has to be moved either to the wish (correct) list or into the rejecting list.

II. Tesseract

Tesseract is an open-source OCR appliance [2]. In between the years 1984 through 1994 it took its origin at HP. Tesseract, in HP Labs Bristol, this project was initiated as a PhD project and later added with hardware and software plugins and this increased most accuracy of its working in HP Scanners sector. At an early stage of OCR engine business were very poor and once this is added up with the good improvement to work with scanners it took an exponential turn in business and execution. This turn was more motivation for further improvements and updates in its working [2].

After better results in scanners, Tesseract did not materialize into the product. As in the development stage of OCR compression Tesseract came into focus once again and took forward to further developments. This time complete focus was to improve rejection efficiency than on base-level accuracy. This project came to an end in the year 1995 and sent to the UNLV Annual Test of OCR Accuracy. After successful completion of the test it was introduced to the world, showing how efficient and powerful when compared with the commercial engines. Earlier this was not an open source, but from late 2005 HP released it as open source. Currently it is available at http://code.google.com/p/tesseract-ocr[2].

2.1 Architecture

The Later HP lab developed layout analysis technology independently. The technology was used in products. (And therefore not released to open-source) Because of this technology developed by HP, Tesseract never met the need of its own page layout analysis. Tesseract assumes that its input is a binary image with optional polygonal text regions defined.

Processing follows a traditional step-by-step pipeline, but some of the stages were strange and found very new in those days, and possibly remain so even now. The first step of processing is a connected component analysis. In this step, the outlines of the components are stored. With regard to computation, the design decision was an expensive one at the time. But it had a significant advantage: by inspecting the nesting of outlines, and the number of child and grandchild outlines, the detection of inverse text became simple and recognition of the text became very easy in recognition of black-on-white text. Tesseract is probably the first OCR engine that is capable of handling white-on-black text so a little effort. At this stage, outlines are made together, purely by nesting, into Blobs. Blobs are unionized into text lines, and analysis of the lines and regions for fixed pitch or proportional text is done. Based on the kind of character spacing, text lines are broken into words. The character cells chopped the fixed pitch text immediately. Proportional text is broken into words by employing the definite spaces and fuzzy spaces.

Then recognition proceeds as a two-pass process. In the first pass, recognition of each word in turn is attempted. Each word that is satisfactory is passed to an adaptive classifier as a training data. Then the adaptive classifier gets a chance for recognizing the text lower down the page with still more accuracy. Since the adaptive classifier may have learned something useful too late to make a contribution near the top of the page, a second pass is run over the page. In the second pass, the words which were not properly recognized are made to recognize once again, thereby lead to greater accuracy of recognition.

A final phase resolves fuzzy spaces, and checks alternative hypothesis for the x-height to locate small-cap text.

2.2 Line and word finding

2.2.1 Line Finding

There are more publications earlier on line finding, which is one of the major parts of Tesseract. The main application of this algorithm which is designed for Tesseract is to classify skewed pages. This process is more concerned about line construction and blob fitting. The line finding algorithm is one of the few parts of Tesseract that was published earlier. This algorithm is designed for recognizing a skewed page without having to de-skew. This eventually protected the image quality. Blob, blob filtering and line construction are the key parts of the process. As the roughly uniform text size, the drop-caps and vertically touching characters will be removed by a simple percentile height filter with an assumption of page layout analysis. The text in the region is judged by the height of a median. Because of this, filtering out the blobs that are petite then some portion of the median height, for example diacritical marks, punctuation and noise is safe.

The filtered blobs that are more precise to fit a model of non-overlapping, parallel and sloping lines. If the blobs are sorted and processed by the x-coordinate, skew becomes viables in some conditions like assigning the blobs to a unique text line, across the page during the process of tracking the slope, with heavily subsided danger of assigning to an incorrect text line in the existence of skew. After assigning of the filter blobs to lines is done, at least median of squares fit is used to appraise the baselines, Pertinent lines are once again fitted with a filtered-out blobs.

The final step of the line creation process to connect blobs that overlap by at least half horizontally, putting diacritical marks together with the correct base and correctly relating parts of some broken characters.

2.2.2 Baseline Fitting

Once the text lines have been detected, the baselines are fixed in more accurately using a quadratic spline. This was very first done for an OCR system, and enabled Tesseract to treat pages with curved baselines, which are a bit familiar artifact in scanning, and not just a kind of book bindings.

The blobs are portioned into groups with a reasonably continuous displacement for the original straight baseline. This is how the baselines are fitted. A quadratic spline is fitted to the partition which is found to be the most populous (assumed to be the baseline) by a least square fit. The quadratic spline has both advantage and a disadvantage. The advantage being “this calculation is reasonably stable” and the disadvantage being “discontinuities can arise when multiple spline segments are needed”. A more regular cubic spline might work much better [2].

Fig 2.2: An example of curved fitted baseline [2]

Line of text with a fitted baseline, descender line, mean line and ascender line are illustrated in the above fig. All the lines shown in the image are "parallel" as their y separation is a constant for the entire length. Also, these lines are slightly curved. It can be observed in the image that the ascender line is cyan (prints as light gray) and the black line present about this cyan line is actually straight. A deep observation of these lines makes it clear that the cyan (printed gray) line is curved relative to the straight black line above it.

2.2.3 Fixed Pitch Detection and Chopping

The text lines will be tested by Tesseract for determining whether they are fixed pitch text. On the other hand, Tesseract cop's the words into characters using the pitch and also disables the chopper and associator on these words for the word recognition step. A typical example of a fixed-pitch word is illustrated in below fig.

Fig 2.3: A fixed-pitch chopped word [2]

2.2.4 Proportional Word Finding

Non-fixed-pitch or proportional text spacing is a highly non-trivial task. Observe the Fig.3 which illustrates some typical problems. The gap present between the tens and units of 11.9% is similar in size to the general space, but this is certainly more compared to the kerned space present between 'erated' and 'junk'. No horizontal gap exists at all between the bounding boxes of 'of and financial'. Several of such problems are solved by Tesseract by measuring gaps in a limited vertical range existing between the baseline and mean line. At this stage, the spaces that are close to the threshold are made fuzzy, for making a final decision after word recognition [2].

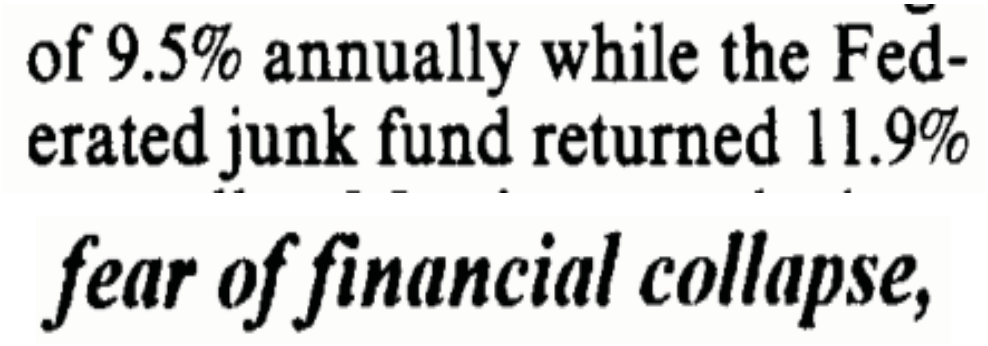


Fig 2.4: Some difficult word spacing [2]

2.3 Word Recognition

For any character recognition engine, the recognition process has to identify the way a word should be segmented into characters. First, the classification of the initial segmentation output from line finding is done. The remaining part of the word recognition step is applicable to non-fixed-pitch text only.

2.3.1 Chopping Joined Characters

While the result of a word from Linguistic Analysis which will be discussed very soon is unsatisfactory, Tesseract makes an attempt for improving the result by chopping the blob with worst confidence from the character classifier. Candidate chops points are found from concave vertices of a polygonal approximation of the outline, and may have either different concave vertex opposite, or a line segment. So as to separate joined characters from the ASCII set successfully, up to 3 pairs of chop points might be needed.

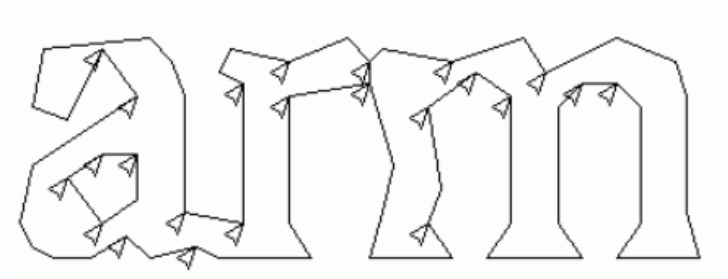


Fig 2.5: Candidate chop points and chop [2]

Above fig shows a set of candidate chop points with arrows and the selected chop as a line across the outline where the 'r' touches the 'm'.

Based on the order of priority of the chops, they are executed. If any chop failed in improving the confidence of the result, it is simply undone, but not completely discarded. It is because; the associator can reuse the chop if it was needed later on.

2.3.2 Associating Broken Characters

If the potential chops have been exhausted, and the word is still not good enough, it will be given to the associator, which then performs an A* (best first) search of the segmentation graph of all the possible combinations of the maximally chopped blobs into candidate characters. It is done not by actually building the segmentation graph, but by maintaining the hash table of the visited states. The A* search is preceded by acquiring candidate new stats from a priority queue and then evaluating them through classification of an unclassified combination of fragments. [2]

There are arguments that this fully-chop-then-associate is, at best it is inefficient and at worst, it is liable to miss important chops and that may well be the case. The benefit is, the chop-then-associate scheme will simplify the data structures that would be required for maintaining the full segmentation graph.

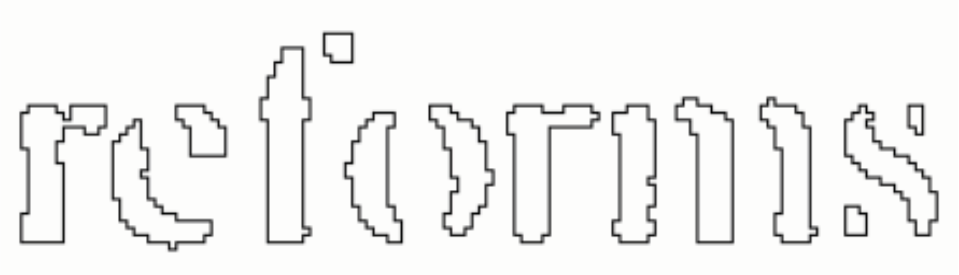


Fig 2.6: An easily recognized word. [2]

When the A* segmentation, search was first implemented in about 1989, the accuracy of Tesseract on broken characters was found to be clearly more than any commercial engine existing at the time. The

above figure is a typical example. The success of Tesseract was because of an essential part called character classifier that could easily recognize is broken characters.

2.4 Static Character Classifier

2.4.1 Features

The earlier version of Tesseract used topological features which are developed from the work of Shillman. In spite of the fact that these elements are autonomous of text style and size, they are not robust to the problems being experienced in real time images. The very next thought that can crack to fix this was to make use of segments of the polygonal approximation as features. Unfortunately, this approach is also not robust to the broken characters. For example, in below figure 2.7 (b) it can be observed that the right side of the shaft is in two main pieces, but in figure 2.7 (a) just a single piece is noticed.

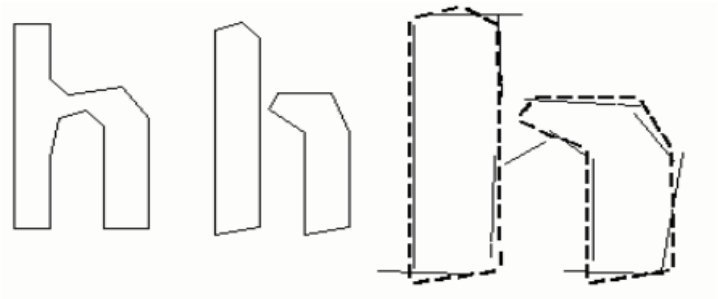


Fig 2.7: (a) Pristine ’h’, (b) broken ‘h’ (c) features matched to prototypes [2]

Then a sudden and dramatic solution came up. It is the idea that the features in the unknown and the features in the training data both need not be the same. In the training process, the segments of a polygonal approximation are used for features, whereas in the case of recognition, extraction of features of a small, fixed length of the outline is done and they are matched many-to-one against the clustered prototype features of the training data. The short, thick lines observed in the above figure, are the features which are extracted from the unknown, whereas the thin and long lines in the same figure

are the clustered segments of the polygonal approximation used as prototypes. One prototype bridging the two pieces is completely unmatched. All the prototypes and every feature are well matched except for the three features on one side and two on the other. This exemplifies that the process of small features matching with the large prototypes is easily able to cope with recognition of the images that are damaged. Its prime downside was the cost of computing the distance between an unknown and a prototype is quite high [2].

Thus, the features that are extracted from the unknown are 3-dimensional, i.e., has x position, y position and angle with typically 50-100 features in a character, whereas the prototype features are 4-dimensional, i.e., x position, y position, angle and length with typically 10-20 features in a prototype configuration.

2.4.2 Classification

Classification is a two-step process, wherein the first step is the creation of a short list of character classes that the unknown might match. This is done by a class pruner. From a coarsely quantized 3-dimensional lookup table, every feature will fetch a bit-vector of the classes that it might match. These bit-vectors fetched are then summed over all the features. The classes with the highest counts (after correcting for expected number of features) become the short list for the next step.

Every feature of the unknown searches a bit vector of prototypes of the given class which it might match with. Then computation of the actual similarity between them is done. Every prototype character class is represented by a logical SOP (sum-of-product) expression with each term. It is called a configuration. A record of the total similarity evidence of every feature in every configuration, and also in each prototype is maintained by the distance calculation process. The best combined distance, which is calculated from the sum of feature and prototype evidences, is the best over all the saved configurations of the class.

2.4.3 Training Data

The classifier is capable of recognizing the damaged characters with a great ease. So, the classifier was not trained on damaged characters. The classifier was in fact trained over a few samples, i.e., 20 samples of 94 characters from 8 fonts in a single size, but with 4 attributes (normal, bold, italic, bold italic). All these sum up to a total of 60160 training samples. This is a significant contrast when compared to other published classifiers, like the Calera classifier which is trained on more than a million samples and the Baird's 100-font classifier which is trained on 1175000 training samples [2].

2.5 Linguistic Analysis

Tesseract has comparatively less linguistic analysis. Every time the word recognition module is considering a new segmentation, the linguistic module selects the best one among all the word strings available in each of the following categories: Top frequent word, Top dictionary word, Top numeric word, Top UPPER case word, Top lower case word (with optional initial upper), Top classifier choice word. For a given segmentation, the final decision will be the word which has the lowest total distance rating, where every category is multiplied by a different constant.

The number of characters in the words of different segmentations may be different. Direct comparison of these words is hard even where a classifier claims to be producing probabilities, which are not produced by Tesseract. This problem in Tesseract is solved by generating two numbers for every character classification. The first one is called the confidence and is minus the normalized distance from the prototype. This enables it to be a "confidence" in the sense that greater numbers are better and the greater the distance from zero, the greater the distance. The second output is called the rating. Its action is to multiply the normalized distance from the prototype by the total outline length in the unknown character. As the total outline length for all the characters within a word will be the same always, ratings for characters within a word can be summed meaningfully.

2.6 Adaptive Classifier

It was suggested and even demonstrated that the use of an adaptive classifier will benefit the OCR engines. The static classifier must be good at generalizing to any kind of font. As a result of which the ability of the classifier to discriminate between different characters or between characters and non-characters weaken. So, for realizing a greater discrimination within each document, where a less number of fonts are existing, a more font-sensitive adaptive classifier trained by the results of the static classifier is commonly used [2].

Template classifier is not used by Tesseract, but it uses the similar features and classifiers of static classifier. Static classifier will normalize the characters by the centroid (first moments) for position and second moments for anisotropic size normalization.

The normalization of baseline/x-height will can easily differentiate upper and lower case characters. It will also improve the liberty to noise spikes. The major advantage lying behind the character moment normalization is removed of font aspect ratio and some degree of font stroke width. Not only that, it also makes the recognition of sub and superscripts simpler, but needs an additional classifier's feature for marking the difference between some upper and lower case characters. 3 letters in baseline/x-height normalized form and moment, normalized form are illustrated in below figure [2].

Fig 2.8: Baseline and moment normalized letters [2]

III. Text Mining

Most previous studies of data mining have concentrated on structural data, like relational, transactional, and data warehouse data. [3] In practice, a considerably higher amount of the available information is backed up in a text database (or document database), which comprises a large collection of documents from several different sources, like news articles, research papers, books, digital libraries, e-mail messages, and web pages. Inferable from the lofty ascent in the measure of data accessible in electronic structure, for example, electronic distributions, different sorts of electronic records, email, and the World Wide Web, there is a precarious and sudden ascent in the text database. These days, almost all the information on government, industry, business, and other institutions are stored electronically, i.e., in the form of text database [3].

Once information is uploaded in the database, most of this data will be in an unstructured way. In recent times we find much research is focused on semi structured data. Additionally, data recovery strategies, for example, indexing techniques, have been created to handle unstructured data.

Conventional information retrieval procedures get to be deficient for the inexorably boundless measure of text data. Normally, just a little portion of the numerous accessible documents will be applicable giving individual client. Without recognizing what could be in the documents, it is hard to define viable inquiries for analyzing and obtaining required information from data [7]. Users need tools to compare different documents, rank the documents, or find patterns and trends across multiple documents. Thus, text mining has become a major theme in data mining.

Information retrieval (IR) and Database systems were developed parallel from the recent past. IR deals with the organization and retrieval of information with huge datasets mostly text based information. Whereas, database systems has fixated on query and translation processes of structured data. Both handles various types of information, some database system issues are generally not exhibit in a data

retrieval system, for example, concurrency control, recuperation, exchange administration, and redesign. In traditional database systems, we cannot find any common information retrieval issues, like comparative search based on key words and unstructured documents. The major issue in information retrieval system is to trace a document in a complete set of collecting data based on a client's query. In this issue user has a handful of choice to pull the relevant information from the stored data, this fits best if the user has same ad-hoc information need, an IR system can also take action to push a newly found data item to a user if it is similar to a client's query. Such an information access process is called information filtering, and the system used in this process is called filtering systems or recommender systems.

2.1 Text Retrieval methods

Text retrieval methods are majorly classified into two

a) Document selection problem

b) Document ranking problem

The document selection method is the one in which the query is the specifying constraint for selecting the relevant document. The boolean retrieval method is one such method in which a set of key words represents a document and a Boolean of expressions provided by the user like tea or coffee and car and shop. All it takes is the Boolean query and return document for this system to satisfy the Boolean expression. Prescribing user information with a Boolean query is really difficult and this retrieval method works well only when the document collection is well known to the user and a good query could be made out of it.

The document ranking method based on the order of relevance is used to rank the documents. These are more significant than the selection methods for there are more advanced retrieval systems present,

which provide a rank list of the documents. Algebra, logic, probability, and statistics are various mathematical tools on which many ranking methods developed. The instinct behind all these methods is to match the key words in all the documents and score them based on how their matches. The degree of relevance of a document with the score based on the information like frequency of words in the document and to approximate it is the main goal. To measure the degree of relevance it is very difficult.

2.2 Cluster Analysis

The process of grouping a set of physical or abstract objects into classes of similar objects is called clustering [4]. A cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the object in other clusters [5]. Classification plays major role in differentiating groups or classes of objects. This is done with the prior knowledge of training tuples or big datasets or different kinds of patterns, in which classifier can model each group. It is often more desirable to proceed in the reverse direction: first partition the set of data into groups based on data similarity (e.g., using clustering), and then assign labels to the relatively small number of groups. Extra points of interest of such a clustering based procedure are that it is versatile to changes and bail single out helpful features that recognize diverse groups. By automating clustering, we can identify dense and sparse regions in object space and, therefore, discover overall distribution patterns and interesting correlation between data attributes [6].

Based on applications, clustering can also be defined as data segmentation as it can classify large data sets into groups according to their similar features. Clustering can also be used for finding data away from boundaries or cluster threshold values as this may also be a point of interest from clients query. As of now Data clustering is under rapid improvement. Contributing territories are examined incorporate data mining, statistics, machine learning, spatial database technology, science, and marketing. From the tremendous measures of information gathered in a database, clustering is the most active topic of

interest in data mining research. As a branch of statistics, cluster analysis has been extensively studied for long time period, which concentrate mainly on distance-based cluster analysis. A cluster analysis tool based on k-mean, k-medoids, and several other methods have also been built into many statistical analysis software packages or systems, such as S-Plus, SPSS and SAS [3]. In machine learning, clustering is an example of unsupervised learning. As it tries to learn from observations instead of examples. More efforts were observed in finding best, effective and efficient cluster analysis methods in the area of data mining for huge databases. Scalability of clustering was one of the most prominent factors to obtain [7].

2.2.1 Categorization of Major Clustering Methods

There exist many clustering algorithms in literature. Classifying the clustering methods as the methods may overlap so that the features from several categories are obtained. The categories of major clustering methods are classified into:

A) Partitioning Methods

In this method if we consider n as objects of data tuples and K as the number of partitions, then K<=1 represents the cluster formed, this formed cluster should fulfill below requirement:

a) Each group must have at least one object

b) Each object must belong to exactly one group [8]. Partitioning gets started with a method called partitioning method which can be improved by iterative relocation techniques. This technique delivers object in the same cluster, which are imminent to each other. In K-mean algorithm cluster is defined by the mean value of objects. These methods find important in finding spherical shaped clusters in small to medium sized databases.

B) Hierarchical Methods

A hierarchical method is the one which creates hierarchical decomposition of the objects in the given set. This method is categorized into

a) Agglomerative

b) Divisive based on the decomposition.

The agglomerative approach, also called top-down approach in which each object to a separate group. All the objects are combined till all the groups are merged into one. In this method, it is evident that a step can't be made undone. The charges for computation is lesser which does not affect combinatorial number. The approaches for improving the quality of clustering is: to perform analysis of object linkages at partitioning and to integrate agglomeration and other approaches.

C) Density Based Methods:

The distance between the objects lay the basis for partitioning methods. It can only find spherical shaped clusters, but not arbitrary. The general method is to grow clusters until density exists in such that the radius of the neighborhood contains the least number of points so that arbitrary shape clusters could be used.

DBSCAN and extensions are density based methods growing clusters for a density-based analysis. DENCLUE is the method in which the analysis happens to the value of distribution of density functions.

D) Grid Based Methods

These quantize the item space into a constant number of cells which frame a grid structure and all the clustering calculations are done on it. Fastest processing time and dependency only on cells in the quantized space are the advantages. STING is a typical grid based method. Wave Cluster applies wavelet transformation for clustering analysis and is both grid-based and density-based [3].

E) Model-based methods

Model-based methods will hypothesize a model for every one of the clusters and will find the best fit of the data to the given model. A model-based algorithm may find the location of a cluster by constructing a density function which reflects the spatial distribution of the data points. It also results in a way of determining the number of clusters automatically on the basis of standard statistics, taking "noise" or outliers into account. Hence it yields in robust clustering methods.

EM is an algorithm which performs expectation-maximization analysis depending on the statistical modeling. COBWEB is a conceptual learning algorithm. It performs probability analysis and takes concepts as a model for clusters. SOM (or self-organizing feature map) is a neutral network-based algorithm which clusters by mapping high-dimensional data into a 2-D or 3-D feature map, which is in turn used for visualizing the data. The selection of clustering algorithm is done based on the type of data that is at hand and also the specific purpose of the application [3]. If a cluster analysis is employed to serve as a descriptive or exploratory tool, there are possibilities for trying numerous algorithms on the same data for seeing what the data may disclose.

A few of the clustering algorithms combine the ideas of different clustering methods, as a result of which the classification of a given algorithm as uniquely belonging to only one clustering method category becomes difficult. On the other hand, there are a few applications which may have clustering criteria which need the integration of several clustering techniques. In addition to the

above mentioned categories of clustering methods, there are two more classes of clustering tasks which need special attention. One of them is clustering the high-dimensional data, and the other one is constraint-based clustering.

F) Clustering high-dimensional data

This is particularly a task of the highest importance in cluster analysis as several applications need the analysis of objects which have a large number of features or dimensions. For instance, consider the text documents which may possess thousands of terms or keywords as features, and DNA microarray data may give information on the expression levels of thousands of genes under hundreds of conditions. Clustering high-dimensional data has become a challenging task because of the curse of dimensionality. Many dimensions may not be relevant. As the number of dimensions increases, the data become increasingly sparse, so that the distance measured between pairs of points becomes an effort to compute and the average density of points anywhere in the data is likely to be low. Hence a need for developing a different clustering methodology for high-dimensional data exists. CLIQUE and PROCLUS are two influential subspace clustering methods, which search for clusters in subspaces (or subsets of dimensions) of the data, rather than over the entire data space [3]. Frequent pattern-based clustering, another clustering methodology which extracts distinct frequent patterns among subsets of dimensions which occur quite often. It makes use of such patterns of grouping objects and for generating meaningful clusters. pCluster is an example of frequent pattern-based clustering that groups objects based on their pattern similarity.

G) Constraint-based clustering

This is another type of clustering approach which performs clustering by incorporating the user-specified or application-oriented constraints. Each constraint either specifies a user's expectation, or describes "properties" of the desired clustering results, and thereby providing an effective means

for communicating with the clustering process. Different types of constraints can be specified, either by a user or as per the requirements of the application. Our discussion here will be concentrated on spatial clustering with the existence of obstacles and clustering under user-specified constraints. In addition, semi-supervised clustering is described, which applies, for example, pairwise constraints (such as pairs of instance labeled as belonging to the same or different clusters) in order to improve the quality of the resulting clustering. In the following sections, we examine each of the above clustering methods in detail. We also introduce algorithms that integrate the ideas of several clustering methods [3].

IV. Machine Learning

Machine learning is generally taken to encompass automatic computing procedure based on logical or binary operations that learns a task from a set of examples. Here we are just concentrating with classification. Machine learning aims to generate classifying expression simple enough to be understood easily by the human. They must mimic human reasoning sufficiently to provide insight into the decision process. Like statistical approaches, background knowledge may be exploited in developing, but operation is assumed without human intervention [11].

The basic framework for machine learning is depicted in below figure. The learning system aims at determining a description of a given concept from a set of concept examples given by the teacher and from the background knowledge.

Implemented examples can be positive or negative. The learning algorithm then builds on the type of examples, on the size and relevance of the background knowledge, on the representational issues, on the presumed nature of the concept to be acquired, and on the designer's experience. An important requirement is that the learning system should be able to deal with the imperfections of the data. Examples will often contain a certain amount of noise errors in the descriptions or in the classifications.

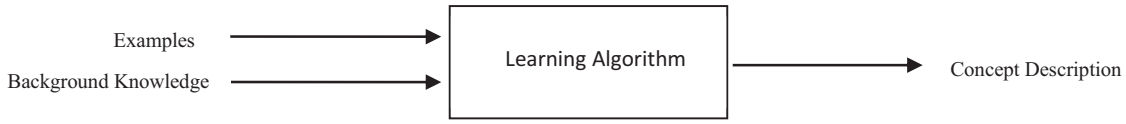


Fig 2.9: Machine Learning Task

There are three different types of learning.

Supervised Learning: In order to predict new values the system has to learn a mapping between the inputs and outputs.

Unsupervised Learning: The features in distributed input data are extracted by the system. Reinforcement Learning: The system has to learn a policy such that each action should result to maximal reward [13].

Fig 2.10: a) Supervised learning b) Unsupervised learning c) Reinforcement learning

2.1 Multilayer Perceptron (MLP)

Covers theorem on separability clearly states that, “If a complex pattern-classification data which cannot be linearly separable in lower dimension when projected into higher dimension than data can be linearly separated. Provided that space is not densely populated.

An MLP composed of Input Layer, several Hidden Layer and Output Layer. Input data is projected into higher dimensions via hidden layer, if this hidden layer is competently chosen, then output neuron can learn to classify the data correctly [12].

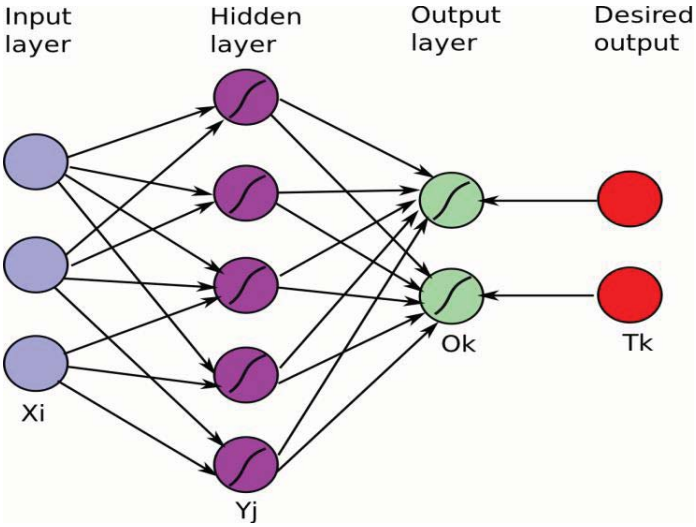


Fig 2.11: Projection of data in MLP

The hidden neuron and output neuron performs a weights sum of their input called net activation and apply the transfer function to this sum to obtain the output. The most used transfer functions are Logistic function, Threshold function, linear function and Hyperbolic function.

The leaning principle in MLP is by Back Propagation. As shown in the fig [2.11], when the output O_k is not equal to the desired output Tk, the data are back propagated via output layer to hidden layer and weights are corrected in the hidden layer such that data is linearly classified. This back propagated error represents the percentage of error in the output layer that is caused by the hidden neuron. The hidden layer is generally used to make a bottleneck, forcing the network to make a model of the system generating the data, with the ability to generalize to previous unseen pattern [13].

Learning in a MLP: The Back propagation Learning Rule

The back propagation algorithm consists of a feed forward pass meant for computing the activities in all layers. The w_{jk} means that the hidden neuron j participated a lot in the error made in the output neuron k.

For the hidden neurons

$$y_j = f\left(net_j(x)\right) = f(\sum_{i=1}^{N_j} w_{ij}.x_i + b_j)$$

For the output neurons

$$O_k = f\left(net_k(y)\right) = f(\sum_{j=1}^{N_k} w_{jk}.y_j + b_k)$$

And of a feedback pass to back propagate the error of the network.

$$\delta_k = (t_k - O_k).f'(net_k)$$

$$\delta_j = f'\left(net_j\right).\sum_{k=1}^{N_k} w_{jk}.\delta_k$$

2.2 Radial Basis Function (RBF)

The RBF neural network is similar to the other net algorithm, but used different error estimation and gradient descent function and result obtained from them is also difficult to understand.

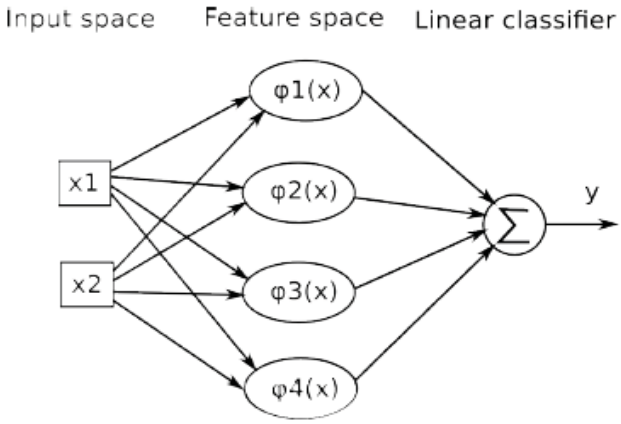


Fig 2.12: Radial Basis Function network

The RBF network consists of a layer of units performing linear or nonlinear functions of attributes. Its structure looks similar to MLP with one hidden layer called as feature space and the input data of dimension d is transformed through φ into feature space. If the function ϕ is chosen correctly the data become linearly separable [13]. The most popular function used is a Gaussian function because of its localized properties as maximal for the center, decreasing to zero for bigger distances. Each hidden unit specializes on a restricted part of the input space, the resulting code is sparse (only a few $\varphi_i(x)$ are different from 0).

The output of linear classifier y is defined as

$$y = F(x) = \sum_{i=1}^{N} w_i \cdot \varphi_i(x)$$

The function ϕ is called Radial-Basis Function, as it is localized around its center and symmetric.

The function used for the hidden neuron is Gaussian centered on each training example

$$\varphi_i(x) = \exp(-||x-x_i||^2)$$

The classification problem can be rewritten as

$$\Phi.w = t$$

Where $\Phi = \{\varphi_{ij} = \varphi(||x_j - x_i||)\}_{i,j=1}^{N}$ is the interpolation matrix, w is the linear weight vector, and t is the desired output.

RBF uses a cross validation technique to handle noise. As error on the cross validation set increases algorithm stops training, which is most advantageous when compared to other algorithms. The hidden layer is computed through a single function ϕ, but not a series of weights as in MLP [13].

The other advantages of RBF when compared with MLP are:

- RBF is faster to use than MLP, as the hidden layer is computed only through a function of distance not weights.
- With the same number of hidden layers, RBF is more precious than MLP in terms of the generalization error.
- RBF performs faster than MLP [13].

Chapter 3 Concept

In this chapter, we discuss more about the concepts, formulas we implemented in this thesis work. In order to speed up the application process, we need to automatize pre-processing stage. The following architecture is suggested for automation process.

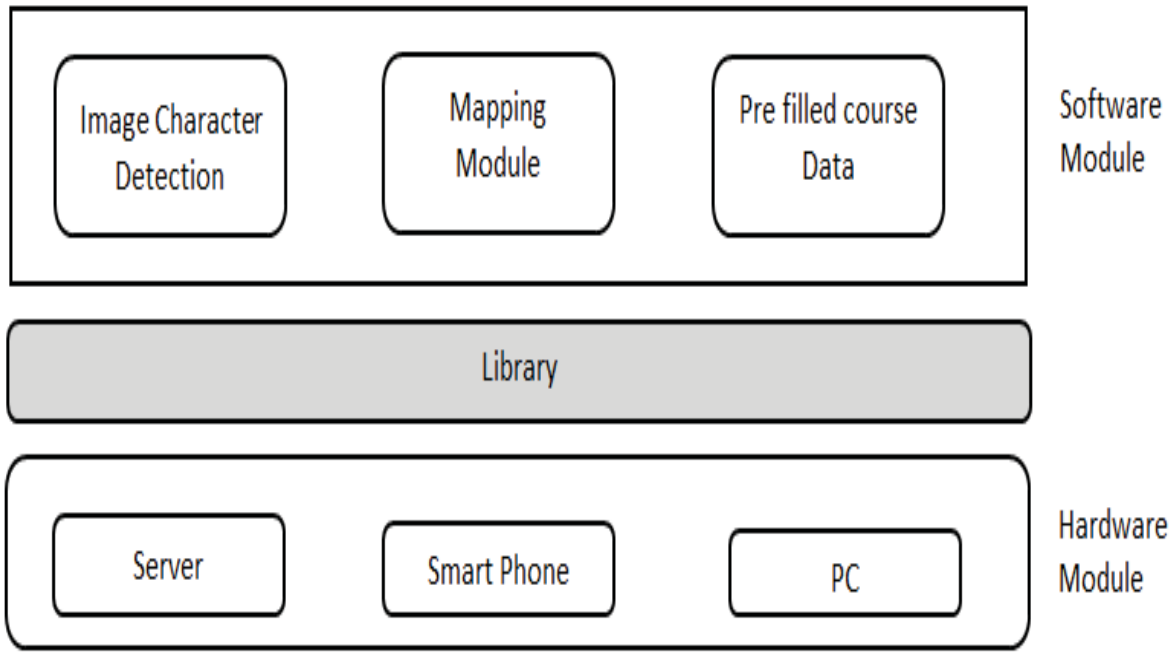


Fig 3.1: Architecture of concept

The complete architecture of the working model is described in the figure 3.1. The model is divided into two major blocks one supports the software part called software module and the one that supports hardware infrastructure named as hardware module. These modules are well connected with the

required library functions. The major components of the software module are Image Detection, Mapping module and GUI and hardware that are used are servers, client PC and mobile.

3.1 Image Character Detection

In this module the image is given as input in any of formats like PDF, tiff, etc., the data in this image or document provided as input should be read and its digital format should be provided as output so that this result can be used for other applications. This conversion of one format to another is done by using OCR.

OCR is a process in which alphanumeric characters in document are converted automatically into computer readable code text. OCR currently encompasses two distinct areas – pure optical character recognition, using optical techniques such as mirrors and lenses and digital character recognition, using scanners and computer algorithms. The field has come far, from an application that require font-specific training, to current, intelligent application that can recognize most fonts with a high degree of accuracy. Most of these intelligent applications use some forms of machine learning to adapt and learn to recognize new fonts. The Current OCR machine achieves an accuracy of more than 97% with computer documents, 80-90% with clearly handwritten text on clear paper. Recognition of cursive text is an active area of research. In this process, there are many sub processes performed for further analysis like the function pre-processing uses the input image takes from scanner or any other source to transfer the original text image into a bitmap file or binary matrix. Then it undergoes a text analysis process to section the text image into lines and characters. Next in this process, we have a recognition process in which the data is manually entered in the database and the output data of the scanned image will be compared.

Initial step in this process start when applicant's data processing is entered and the file with applicant details like personal details, educational qualification, work experience, and language skills is given as the input and details likes university, study course are read and this details are compared with the prefilled data in database and we obtain a list of same university and all course list which are given to OCR to conversion and working of OCR start from here and output generated form the OCR will be Text Fragments. The text fragments are given to further mapping module.

3.1.1 General Classification Controller

This is the main module that process required output. It will be a part of Mapping module. The major work involved includes creating files of required format, conversation of formats to map the formats in the database and servers. As the files are generated from the different tools and software's, the final output will use these files to generate or predict some results from these files so the formatting of this file plays a vital role in this process.

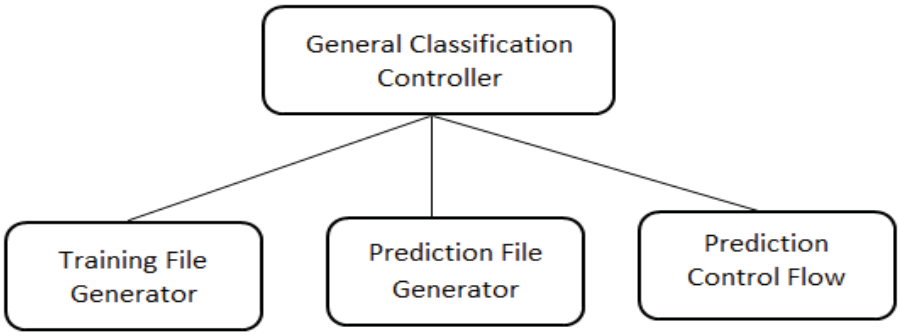


Fig 3.2: General classification segments

The General classification controller comes under mapping module. There are two types of files that are generated in our application when we compare how good SVM and Cluster can classify our input data with a complete course list of university of a particular study path to the courses that can be mapped to ASE related courses. In this module LIBSVM will be triggered on and data training, testing and

classification will be done with it. We implemented SVM in this module in which the input data from OCR and DB prefilled course loader will be used for training SVM. The input data from OCR will be classified in such a way that all text fragments that are generated will be divided into the ASE course list or to exclude list.

The machine learning package used in this thesis is LIBSVM. It is a package for support vector machine technique. LIBSVM includes several kernels, including RBF (radial basis function), a polynomial and a linear kernel. Here for a give training set of instance label pair (x_i, y_i), I = 1, 2, 3…l where $x_i \in R^n$ and $y \in \{1,-1\}^l$, the SVM requires the solution of the following optimization problem:

$$½* W^T *W + C\sum_{i=1}^{l} \xi_i$$

$$\text{Subjected to} \quad y_i (W^T \phi (x_i)+b) >= 1- \xi_i,$$

$$\xi_i >= 0$$

Here training vector x_i are mapped into a higher dimensional space by the function. SVM finds a linear separation hyperplane with the maximal margin in this higher dimensional space. C>0 is the penalty parameter of the error term.

In the below diagram complete working if mapping module is demonstrated. The output of this module will be both SVM and cluster based classification.

SVM Predict:

SVMModel:: Predict – Predict a value for previously unseen data.

Description:

```
public float SVMModel::predict ( array $data )
```

This function accepts an array of data and attempts to predict the class or regression value based on the model extracted from previous trained data. Output will be a class label in the case of classification and a real value in the case of regression [14].

SVM Train:

SVM:: train – creates a SVMModel based on training data.

Description:

```
public SVMModel svm::train ( array $problem [, array $weights ] )
```

Train a support vector machine based on the supplied training data. The output for this will be an SVM Model that can be used to classify previously unseen data [14].

3.2 Mapping Module

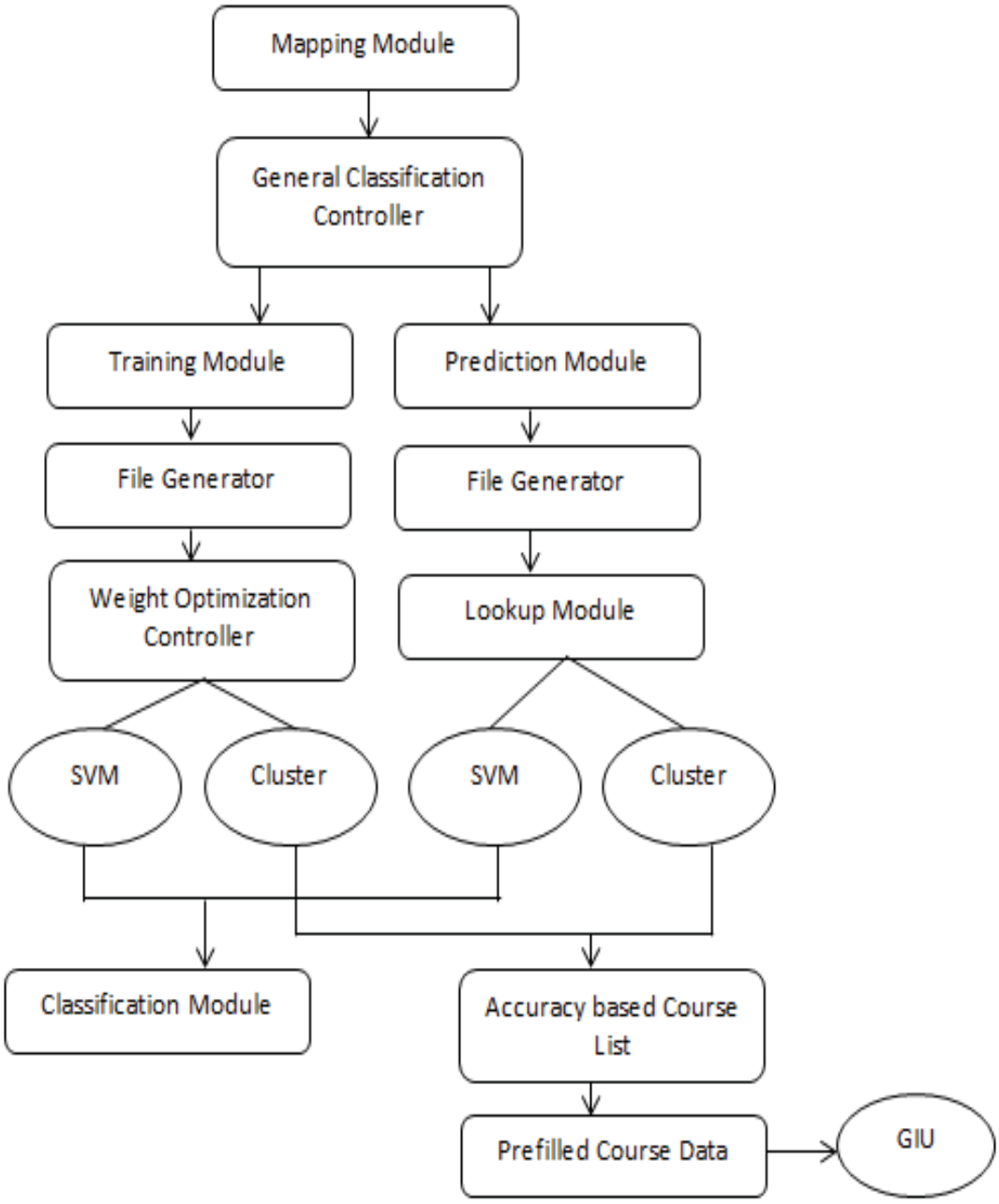


Fig 3.3: Block diagram of Mapping module

The above diagram shows the complete workflow of mapping module, in which SVM and cluster based approaches are calibrated. Tesseract with Java is used to write the complete code structure. Further, in this paper, we discuss about the formation of SVM and Cluster classification.

3.2.1 Weight Optimization and Lookup Module

In this module output of OCR i.e., text fragments is considered and compared with the prefilled courses in database by setting up some parameters while training the SVM or while creating clusters. In case of SVM the process of weight optimization is done while training the SVM data. Once the data are trained, training, file gets generated in this file it keeps counting the characters that are repeated in the OCR text fragments and a parameter value of 0,13 is set for each repetition. This value is compared with the database stored course values and SVM classification on if the course is related to ASE or not is done. This work similar to the lookup module too in which the prediction of SVM is requested, before this request an SVM input string should be generated in which number of characters that are repeated can be generated and a value for this is set.

In clustering module the input for optimization is given from the OCR text fragments, once the training process gets started optimization of character repetition parameter is set, accuracy result is observed in the result file after comparing with the clusters stored in the database. Lookup module starts once prediction request is sent. Here comparison is done with distance measurement and get closest cluster for mapping of courses to ASE.

Distance measurement and border calculations are done with different formulas by taking the length of word into consideration. We do have an output form OCR that can state long lines by considering the certificate format. We observed formats of certificates differs from university to university. The formats can be in text string followed by a number and then again text string to number. This type of format is

considered as long line. For word mapping in clustering, formulas are set for two different types of word strings like long words, words with 4 and 3 characters.

The Long word string is determined by the empty space occurrence in the word string, empty space is given a special parameter value while creating a vector. The formulated way is to find the connectivity, minimum distance, distance and border values. If connectivity is greater than a border then it is mapped to the ASE course. If the words that need to be mapped are with 4 and 3 characters, then they are formulated in a different way, as they are compared with each and every letter in a word one by one and if they are exactly mapped then word with 4 characters are given a bonus of 0,01 to obtain the character cluster preference value and 0,10 of bonus is added to the word with 3 characters.

3.2.2 SVM Classification

SVM are commonly used for binary classification. The basic idea of SVM, introduced by Vapnik in 1995 is to separate two classes with a wide range of margin as possible, assuming that the data is linearly separable. Linear separation can be exemplified by drawing a line on a graph of the training data, separating the two classes, given that the input feature vector is of dimensionaity2. SVM can also be used to classify the data that is non-linearly separable. A cost or penalty parameter may be introduced in order to allow for some misclassification. This C parameter controls the trade-off between allowing for some data points to be misclassified and enforcing the margin between the classes. This can result in so called over fitting, obtaining a more accurate model for that particular set of training data, but one that may not perform well when used with another data set. A lower C value means more misclassification is allowed, perhaps resulting in lower accuracy. It is thus important to select an appropriate value for this parameter in order to achieve good accuracy without over fitting.

LIBSVM is a library for support vector machines. LIBSVM and LIBBINEAR are two popular open source machine language libraries; both of these libraries are developed at the University of Taiwan by Chih-Chung Chang and Chih-Jen Lin. Can be written in Java and C++ languages. Implementation includes following below steps:

- Convert data into SVM format
- Conduct simple scaling of the data
- Consider the RBF kernel K (x; y) = e
- Cross-validation is done to find the best parameter C and gamma
- To train complete training data, we use obtained best values of C and gamma.
- Test

3.2.3 Cluster classification

Both training data and predicted data will be used in clustering. In this clustering module we write a code to calculate minimum distance, distance, border and connectivity. As the data is prefilled in the database using web applications, this data is used for training our module. The parameter values of this data will as distance is 0 and connectivity is 100%. Text fragments obtained from the OCR are given as input that should be mapped to the ASE course path by using the parameters distance, border and connectivity. Here prefilled data is considered as cluster and we try to map the fragments to this cluster and consider word for which we obtain connectivity > border. Cluster approach uses features of distance inaccurate (defuzzificated) calculate way to find closest matching course for giving course fragment. This closeness will be a border for cap calculation.

3.3 Pre filled course Data

Once courses related to ASE study path is generated which is based on the clustering, these courses are loaded to a web page. These are the courses that are mapped to ASE study path. There can be further developments that can be added to this web page, where we can add a button to select or edit the courses and check if all the courses we obtained from cluster classification are correct and if we can use this data for further processing. If we find any disturbed or wrong course we can remove that particular subject from the list by selecting it. We generate a matrix for these courses that are mapped to ASE study path, this will have a good scaling in final result weather to accept the application or not. There are other matrix's like work experience, certifications in computer courses and German language skills that can move the scaling factor bit higher. The OCR detected and ASE study path mapped courses are then presented as preselected input values to the auditor for further processing of the application.

3.3.1 GUI

This is the final component of Software module, which mainly includes data input components and data processing components.

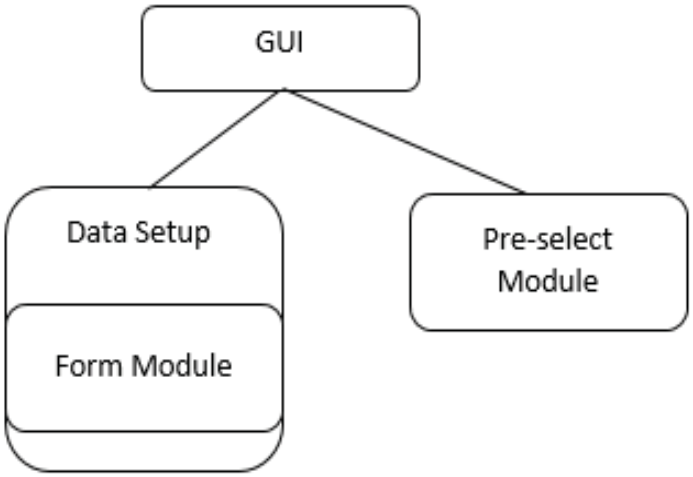


Fig 3.4: Work flow of GUI process

As stated above component to collect data mainly includes manual inputs what auditors do to extract the courses related to ASE study path, updating the database by manual entry of universities from different countries, different study paths from each and every university and finally setting up a database with the courses related to ASE study path. All these upload and updates are done by web applications which are designed for each and every application separately. One other form from which we obtain data is from Uni-assist, in which we have complete data or information about each and every applicant.

The other component which is included in this GUI is a component to process data. The data processing state mainly includes OCR data by which we obtain text fragments, which are in digital format that is sent to Tesseract for further processing. Auditors are also part of this data processing component as they manually update the database and have a look into a complete process of automation. In this process the time saved and effort led by each auditor decreases gradually and efficiency in processing application has increased.

3.4 Library

Library mainly includes the support we get to execute different scripts for all applications in Tesseract. We included different libraries from Java, LIBSVM, Python and PHP, which includes ghost library, JDK and Jar.

3.5 Hardware Module

Hardware module includes, servers, PC and Mobiles. Currently in this project we loaded this tool on server and PC, but farther there can be a scope to develop this project to load this application for mobiles.

Chapter 4
Realization

The below diagram show the flow chart model of code implemented. The complete code is written in Java (Tesseract) initially OCR is called, a Tiff document is given as input to OCR and complete process of OCR as described in the previous chapter is performed and the output from OCR are text fragments. This output is further given to SVM and Clusters and results are compared to check which method works fine to map more accurately to the ASE course list.

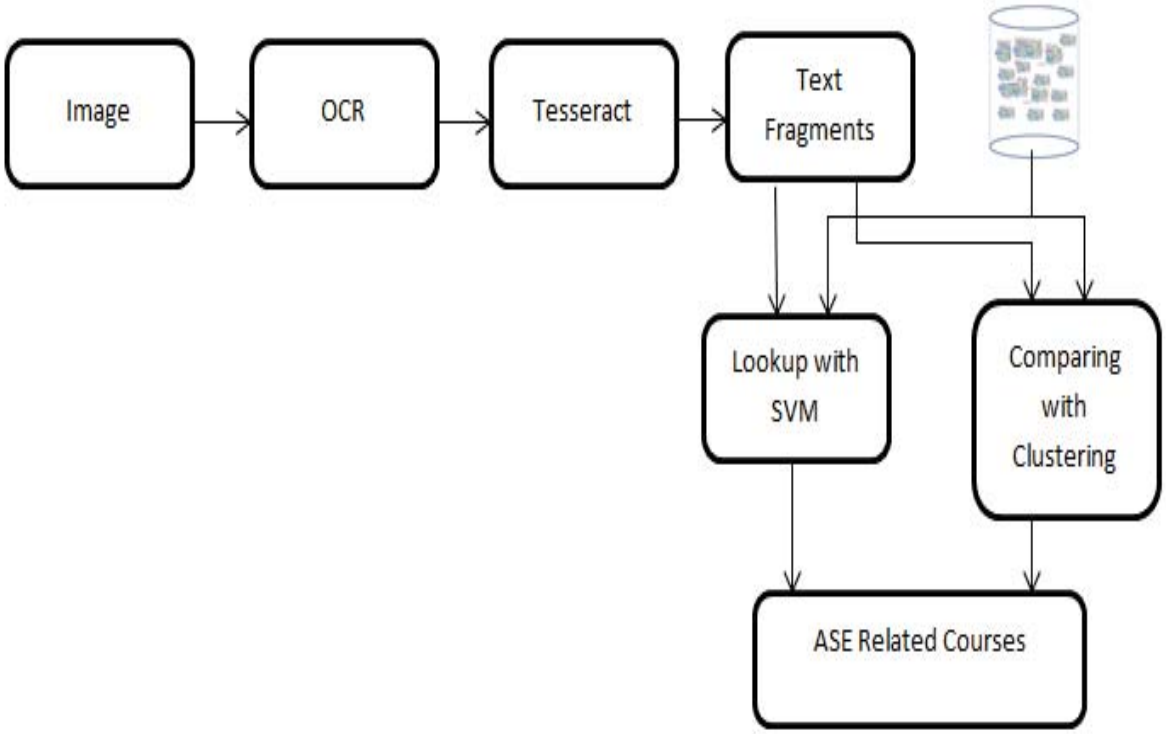


Fig 4.1: Block diagram of complete implementation process

As stated in above figure SVM and Cluster modules are connected with the prefilled database in which we had data about the universities there courses and subjects related to ASE. This data is used as the training our module which is done by using LIBSVM. This training will work effectively with SVM results. In the cluster module we implemented Dimensionality distance calculation for which we used parameters like Distance, Border and Connectivity calculations.

Distance = min. distance / length

Border = 0.1+10*distance

4.1 Web-based Preselect Module

The below figure 4.2 shows a web application by which auditor can enter data manually and upload it to the database for the filtering process. All the applicant data will be displayed on this webpage. Here we have a choice to update the information about new universities, we receive from different countries. Once the details of the applicant are filled and uploaded a unique id is created for each and every application.

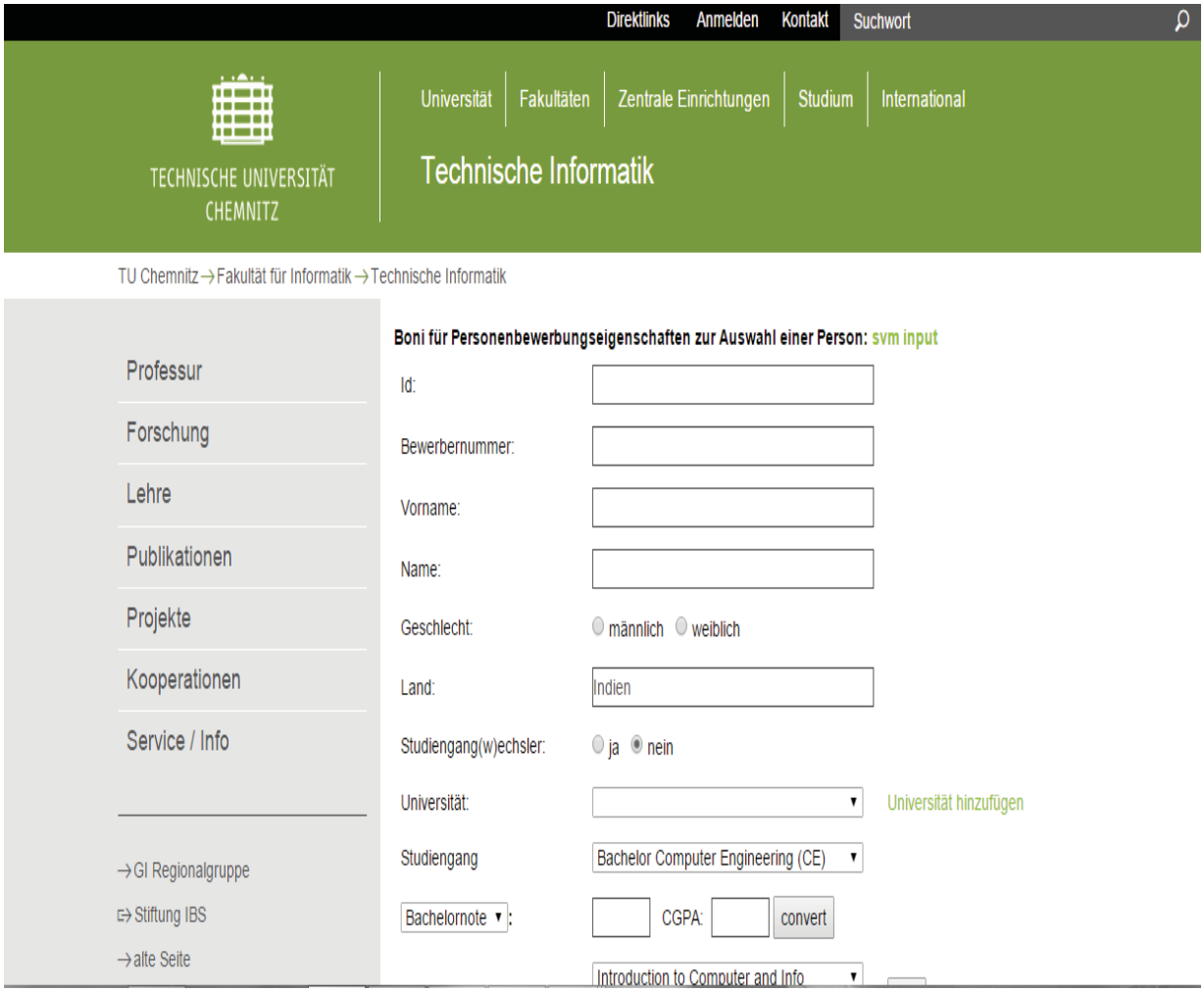


Fig 4.2: Application format we used for complete process

A new option we have included here is SVM input on the top right of the webpage of figure 4.2. Once the data is uploaded and Id is created, some fields which we are interested in are stored in SVM input. This data can be further used in implementing new concepts of SVM in automation process and also to set up high volumes of data samples for training SVM.

The below figure 4.3 shows the web application with an option to update new university. Here we can update the course path and ASE related courses in this path. To the maximum and minimum marks that can be awarded and type of course, like if it is a lab or theory.

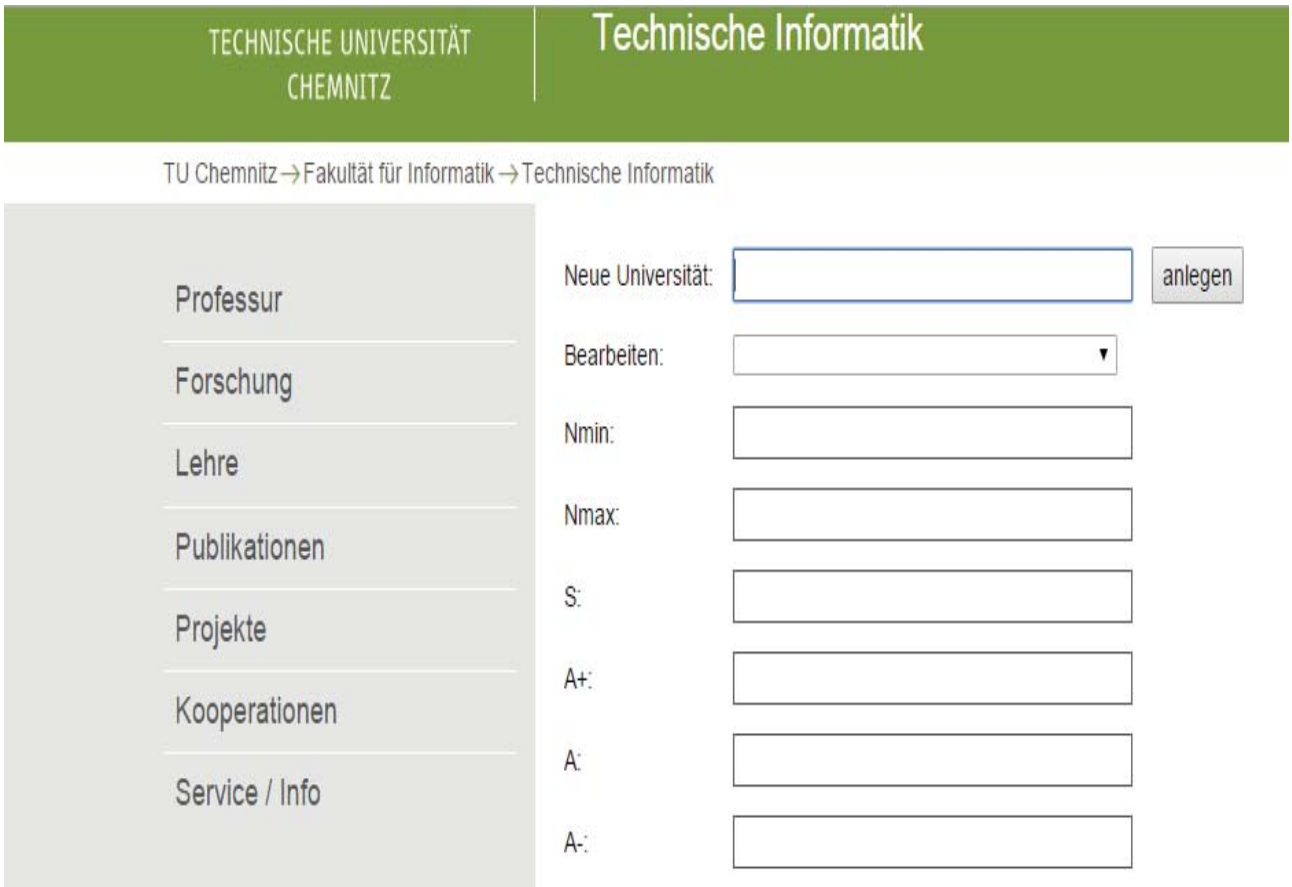


Fig 4.3: Web application to update the universities

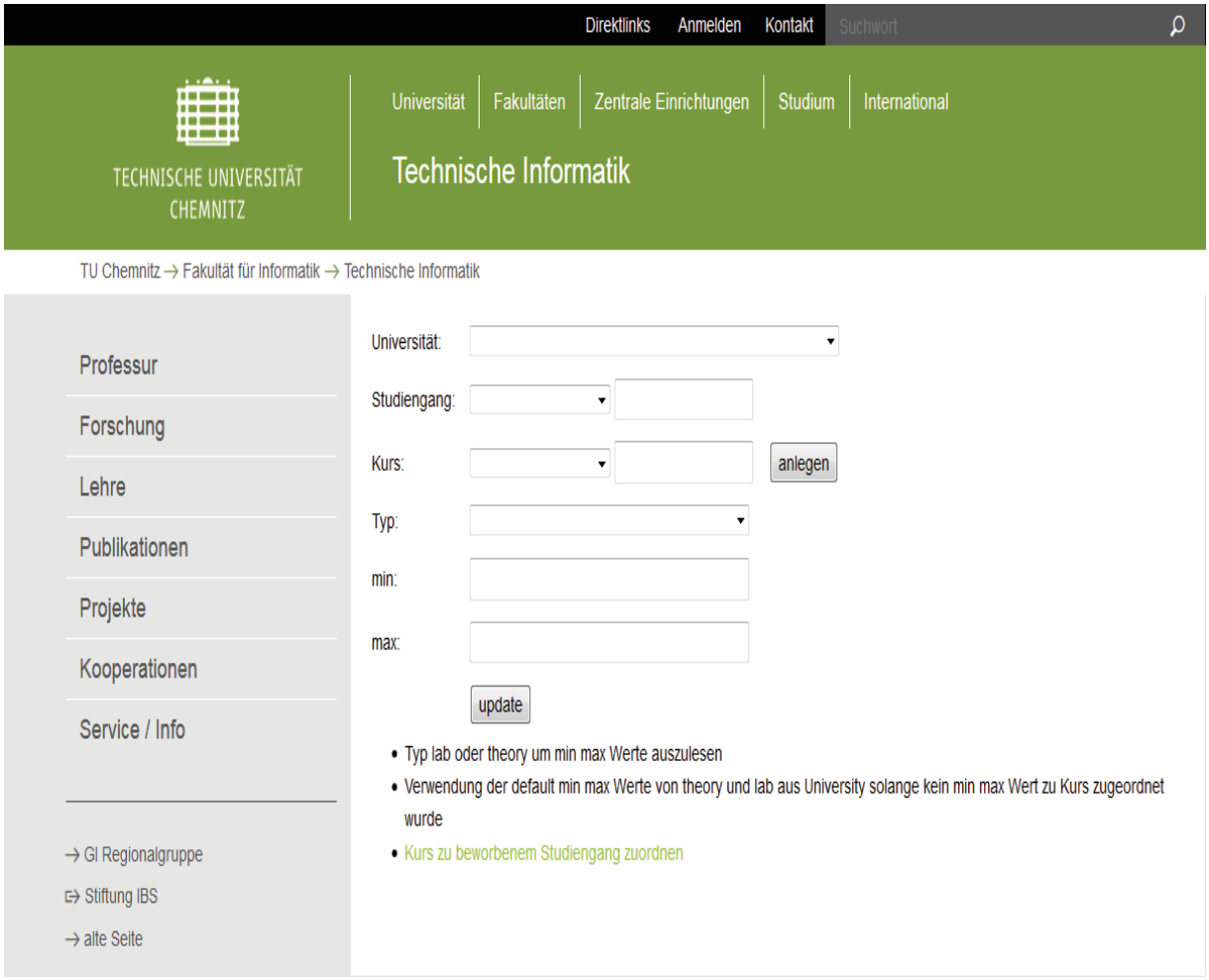


Fig 4.4: Web application to update the courses in universities

As shown in the above figure 4.4, we can update the courses of particular university with this web application. In first tab we can select the university for which we are updating course and we can include type of course it is. It can be a theory or lab and we can award maximum and minimum marks that can be obtained in that course as per university guidelines. This detail can further used to apply a scaling factor such that a general grading that is applicable for all universities can be obtained.

The below figure 4.5 shows the SVM data, we gathered. The first number in a row shows the result and other as the parameter we set in SVM data formats. Here we consider the LIBSVM format as a key value pair. The first number on each line will be a result stating if application is accepted, rejected or application is in process. As stated before SVM data will be in key value pair, key 10 indicates the score obtained by applicants in the previous study course, 11 indicates number of computer related courses, 12 indicates if there are any specialist qualifications, 13 indicates if there are any compute related certificates by the applicant. All these parameters give bonus points to apply for the final selection process.

result accepted(1), not accepted(0); Abschlussnote/6 (10) informatikfächer (11), very good qualifications (12) qualifications (13) other qualifications (14) abbrecher (15)
4 10:0.48817333333333 11:16 12:0 13:0 14:1 15:0
2 10:0.366205 11:22 12:0 13:4 14:1 15:0
2 10:0.46225 11:55 12:0 13:4 14:6 15:0
2 10:0.50576666666667 11:15 12:0 13:2 14:1 15:0
2 10:0.46975 11:17 12:0 13:2 14:0 15:0
2 10:0.48791666666667 11:28 12:0 13:2 14:1 15:0
4 10:0.44381 11:10 12:0 13:1 14:3 15:0
4 10:0.43766666666667 11:9 12:0 13:0 14:6 15:0
2 10:0.23888333333333 11:15 12:0 13:0 14:1 15:0
4 10:0.44391666666667 11:36 12:0 13:5 14:8 15:0
2 10:0.26166666666667 11:27 12:0 13:2 14:1 15:0
4 10:0.39391666666667 11:1 12:0 13:0 14:1 15:0
2 10:0.32166666666667 11:1 12:0 13:0 14:1 15:0
2 10:0.17666666666667 11:1 12:0 13:0 14:1 15:0
4 10:0.502115 11:1 12:0 13:0 14:1 15:0
4 10:0.44058333333333 11:1 12:0 13:0 14:1 15:0
4 10:0.56666666666667 11:1 12:0 13:0 14:1 15:0
2 10:0.34887166666667 11:1 12:0 13:0 14:1 15:0
2 10:0.20891666666667 11:1 12:0 13:0 14:1 15:0
2 10:0.22135 11:1 12:0 13:0 14:1 15:0
4 10:0.36466666666667 11:1 12:0 13:0 14:1 15:0
4 10:0.45181 11:1 12:0 13:0 14:1 15:0

Fig 4.5: SVM data from uploaded applications

Once the courses are entered into database tables, these courses can be mapped to different courses of the master's program. Mapping of bachelor courses to different masters courses is to setups the cluster classification and SVM classification for further developments in this thesis work. This is one new implementation, we made to update the training and cluster data for further projects. In this thesis work we focused on the courses related to ASE so we mapped all the courses to ASE. Mapping of course can be done by button at the end of the webpage and list of courses can be seen in the screenshot.

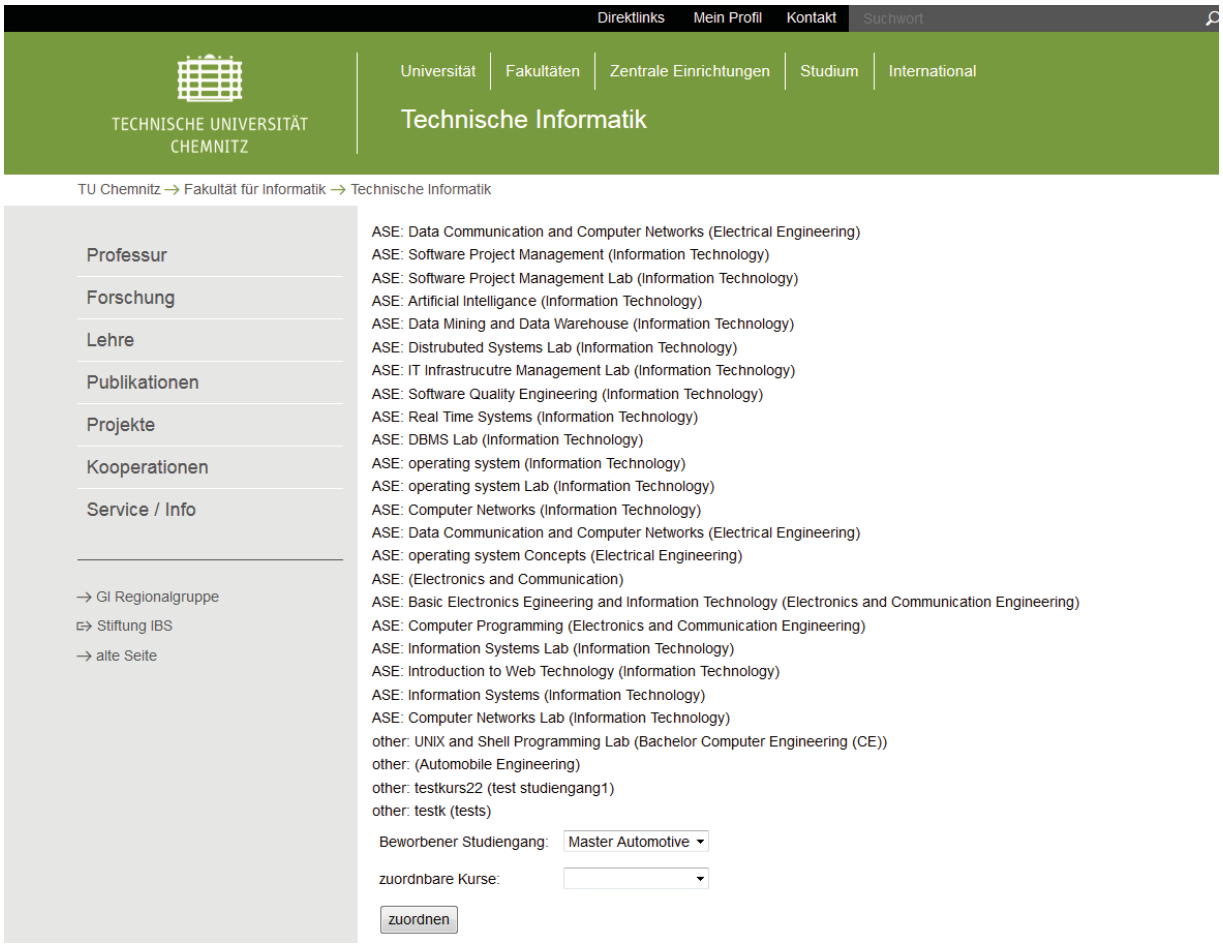


Fig 4.6: Courses mapped to ASE study path

Further, we see the setup of the database and tables that store applicant data. We used PHP My Admin as the database and complete university applicant data is stored in a particular format. In this project we strictly stick to the previous format such that old data we have till now and data we process from now can combine together and setup SVM and clusters accordingly.

The below figure 4.7 shows the table item data of university registration and their Id values. As stated in the above screenshot data entered with university name and the course type, with maximum and minimum marks are stored. These values are further used in the CGP calculation and obtain a particular grade for the selected ASE related courses.

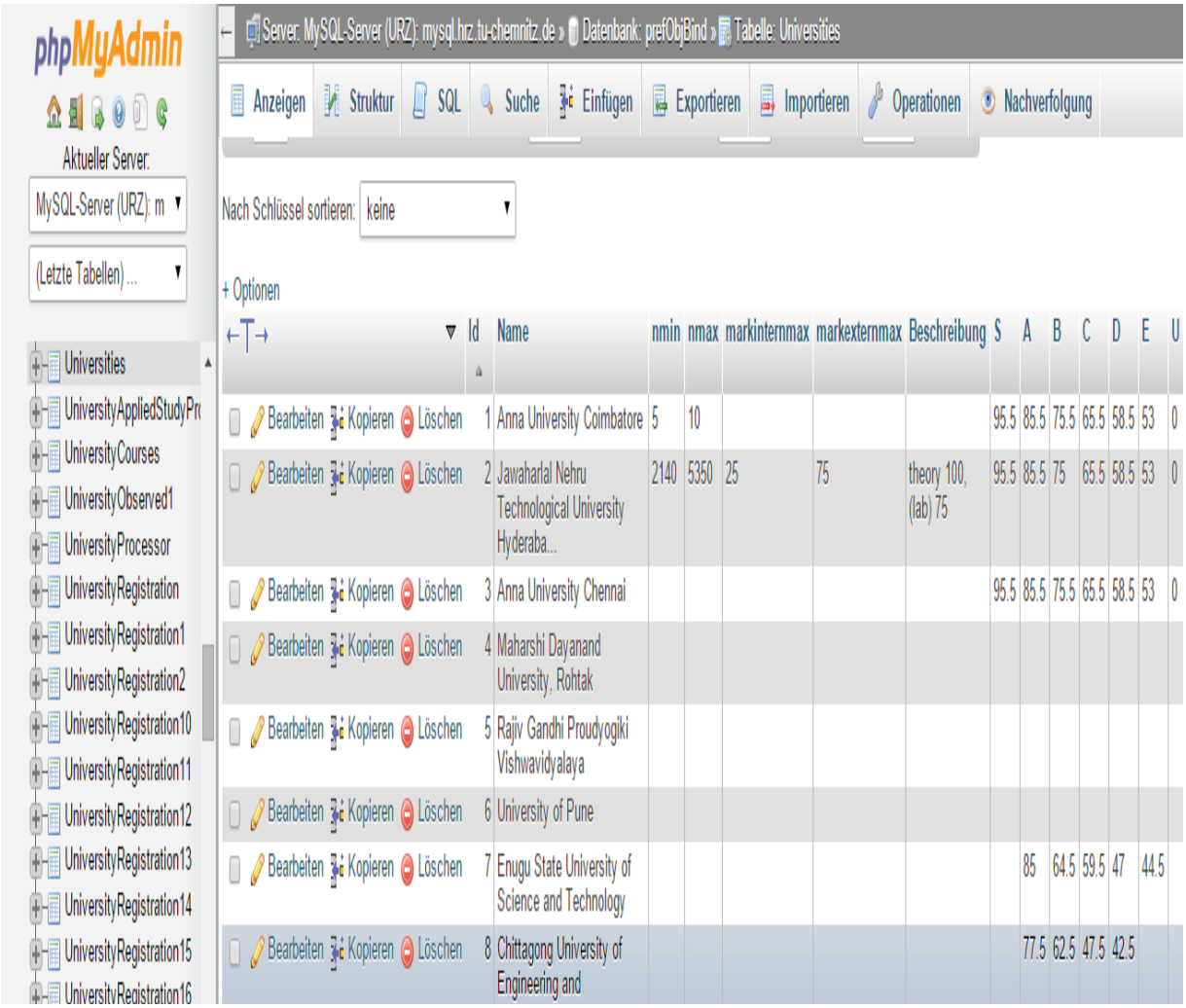


Fig 4.7: Screenshot of tables stored in the database

The below figure 4.8 shows the database tables with UniId that indicates the University, course name with type and study path. These are the courses which are related to ASE study path. Further process followed in this is, if there is an application with this university and course, then the data that are profiled in the below tables are called by Tesseract and mapping of coursed from OCR fragments to these courses are done. Each and every university will have a unique id with respective of the course name.

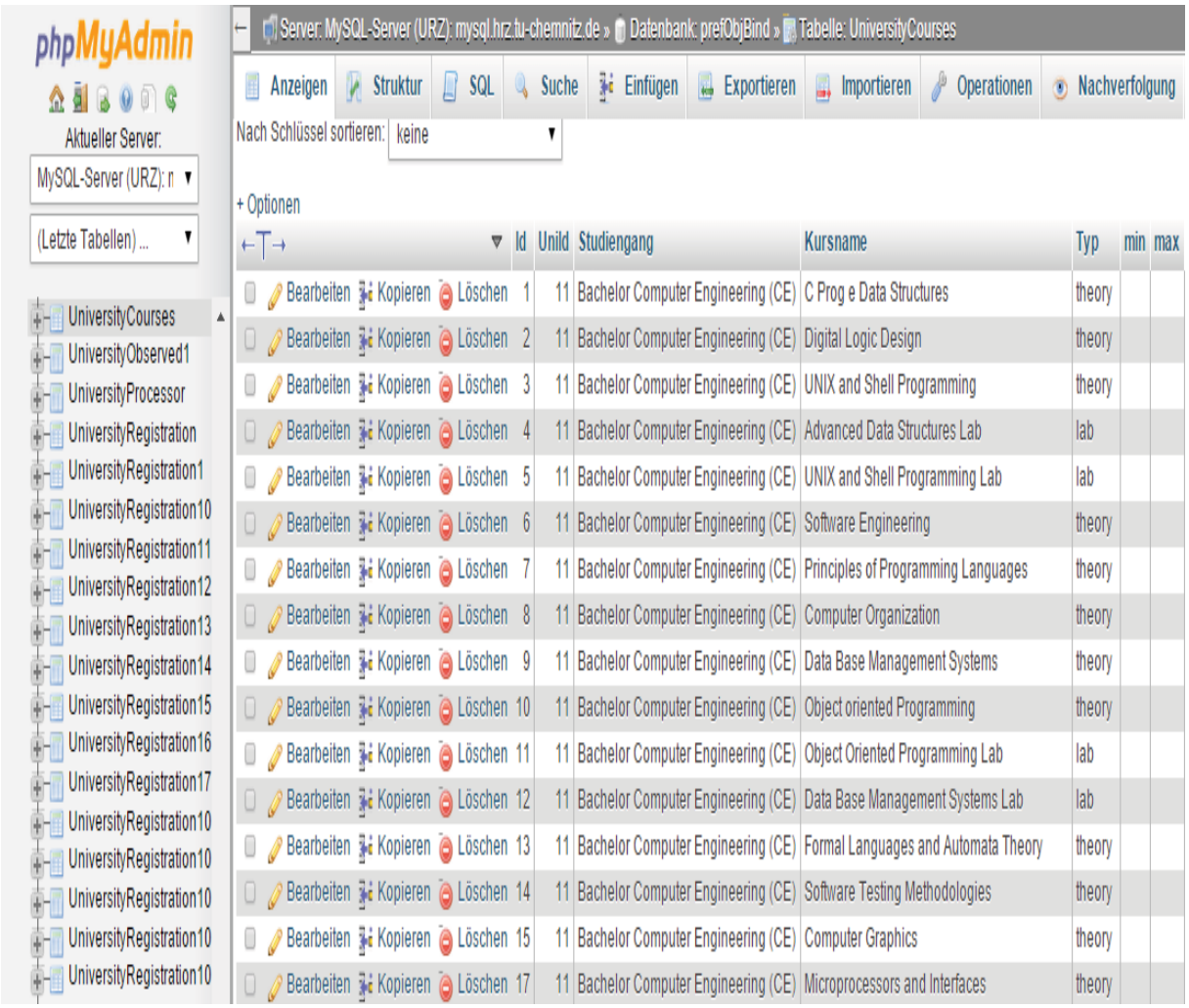
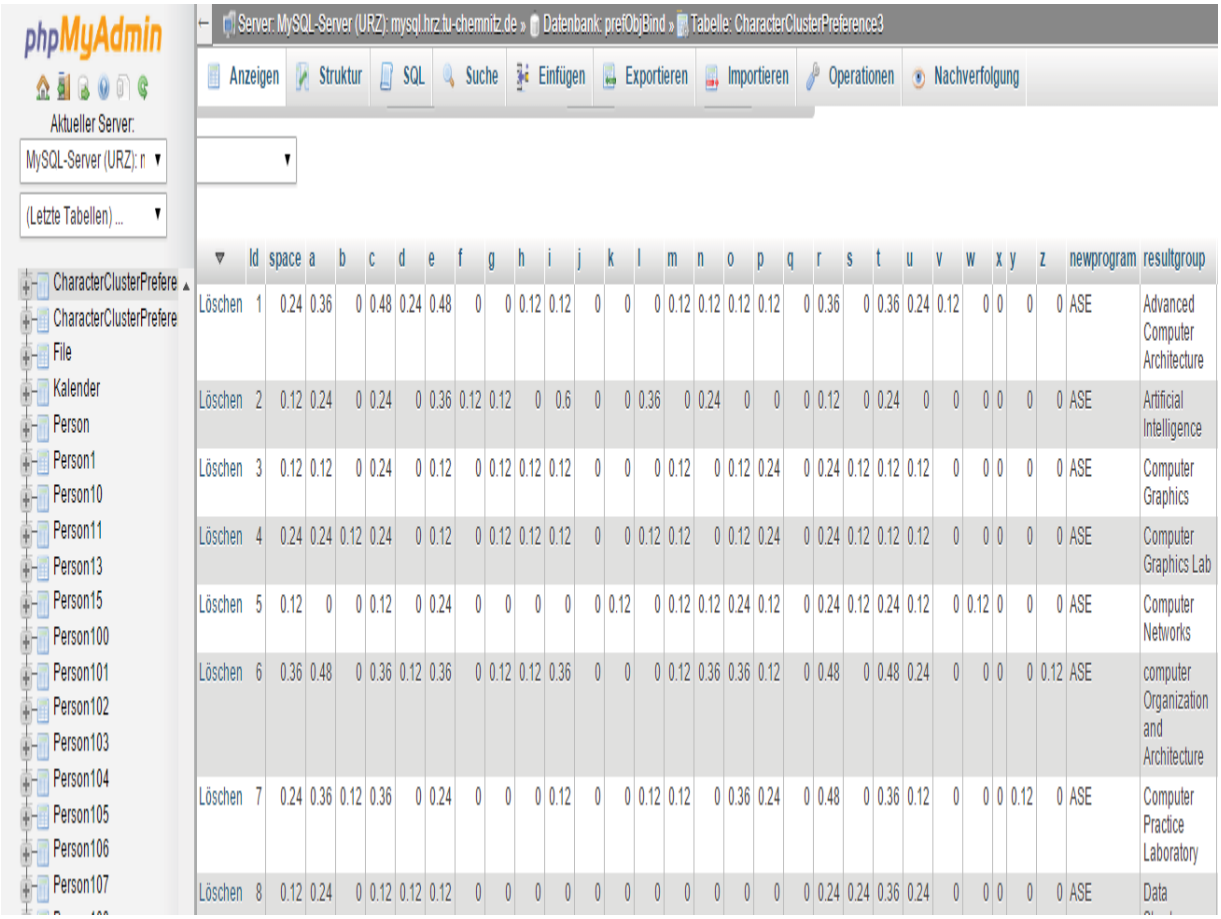


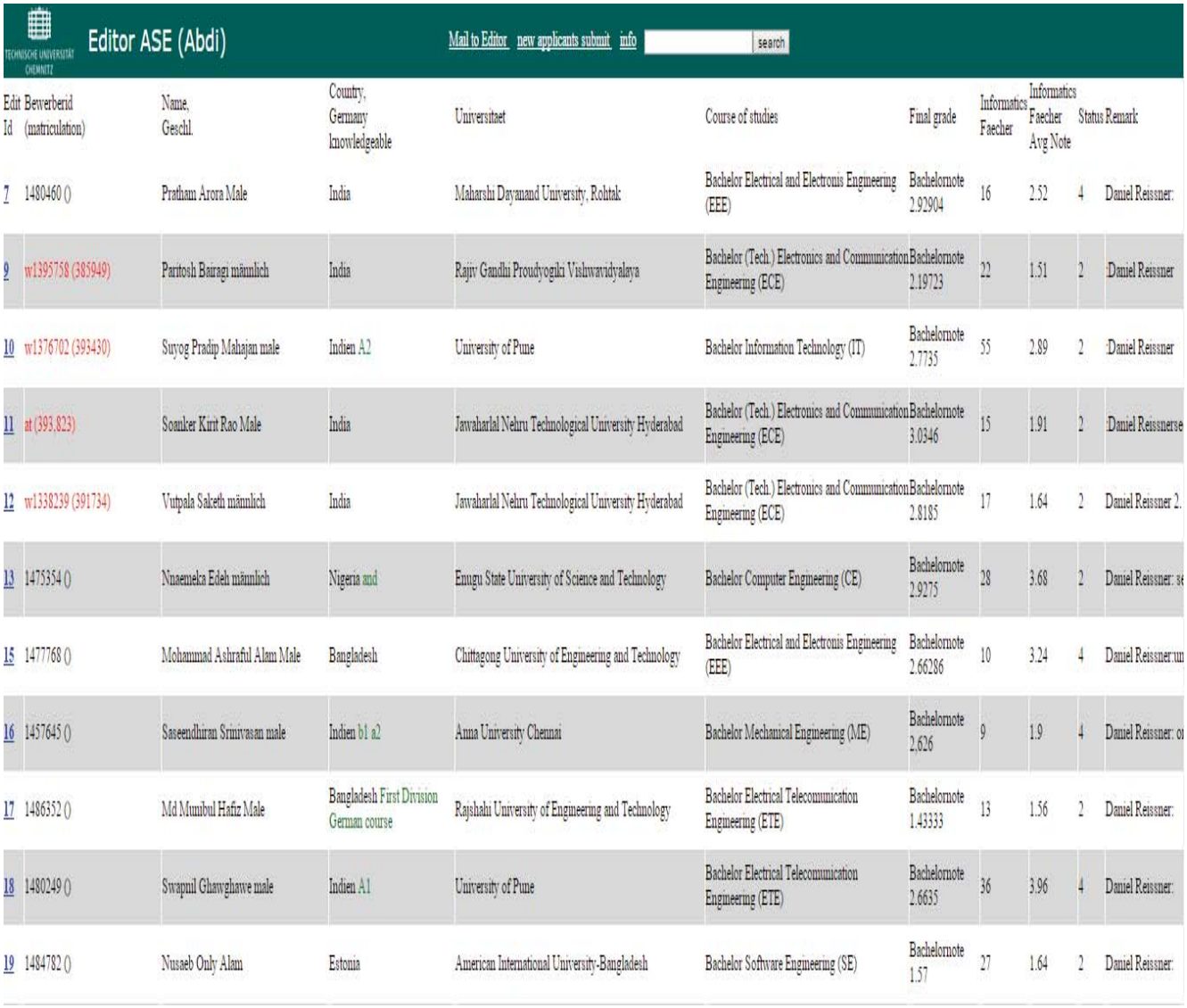
Fig 4.8: University Id and ASE related courses



	Id	space	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	newprogram	resultgroup
Löschen	1	0.24	0.36	0	0.48	0.24	0.48	0	0	0.12	0.12	0	0	0	0.12	0.12	0.12	0.12	0	0.36	0	0.36	0.24	0.12	0	0	0	0	ASE	Advanced Computer Architecture
Löschen	2	0.12	0.24	0	0.24	0	0.36	0.12	0.12	0	0.6	0	0	0.36	0	0.24	0	0	0	0.12	0	0.24	0	0	0	0	0	0	ASE	Artificial Intelligence
Löschen	3	0.12	0.12	0	0.24	0	0.12	0	0.12	0.12	0.12	0	0	0	0.12	0	0.12	0.24	0	0.24	0.12	0.12	0.12	0	0	0	0	0	ASE	Computer Graphics
Löschen	4	0.24	0.24	0.12	0.24	0	0.12	0	0.12	0.12	0.12	0	0	0.12	0.12	0	0.12	0.24	0	0.24	0.12	0.12	0.12	0	0	0	0	0	ASE	Computer Graphics Lab
Löschen	5	0.12	0	0	0.12	0	0.24	0	0	0	0	0	0.12	0	0.12	0.12	0.24	0.12	0	0.24	0.12	0.24	0.12	0	0.12	0	0	0	ASE	Computer Networks
Löschen	6	0.36	0.48	0	0.36	0.12	0.36	0	0.12	0.12	0.36	0	0	0	0.12	0.36	0.36	0.12	0	0.48	0	0.48	0.24	0	0	0	0	0.12	ASE	computer Organization and Architecture
Löschen	7	0.24	0.36	0.12	0.36	0	0.24	0	0	0	0.12	0	0	0.12	0.12	0	0.36	0.24	0	0.48	0	0.36	0.12	0	0	0	0.12	0	ASE	Computer Practice Laboratory
Löschen	8	0.12	0.24	0	0.12	0.12	0.12	0	0	0	0	0	0	0	0	0	0	0	0	0.24	0.24	0.36	0.24	0	0	0	0	0	ASE	Data

Fig 4.9: Generated Character cluster preference

The above figure 4.9 shows the Character Cluster Preference that is generated for each and every university. These cluster tables are filled with the course name and their corresponding parameter values. All the courses which are extracted from a university course are used and parameter values are based on the algorithms we implemented in Tesseract.



Editor ASE (Abdi)

Mail to Editor new applicants submit info search

Edit Id	Bewerberid (matriculation)	Name, Geschl.	Country, Germany knowledgeable	Universitaet	Course of studies	Final grade	Informatics Faecher	Informatics Faecher Avg Note	Status	Remark
7	1480460 ()	Prathaan Arora Male	India	Maharshi Dayanand University, Rohtak	Bachelor Electrical and Electronis Engineering (EEE)	Bachelornote 2.92904	16	2.52	4	Daniel Reissner:
9	w1395758 (385949)	Paritosh Baragi männlich	India	Rajiv Gandhi Proudyogiki Vishwavidyalaya	Bachelor (Tech.) Electronics and Communication Engineering (ECE)	Bachelornote 2.19723	22	1.51	2	Daniel Reissner
10	w1376702 (393430)	Suyog Pradip Mahajan male	Indien A2	University of Pune	Bachelor Information Technology (IT)	Bachelornote 2.7735	55	2.89	2	Daniel Reissner
11	at (393.823)	Soanker Kirit Rao Male	India	Jawaharlal Nehru Technological University Hyderabad	Bachelor (Tech.) Electronics and Communication Engineering (ECE)	Bachelornote 3.0346	15	1.91	2	Daniel Reissnerse
12	w1338239 (391734)	Vutpala Saketh männlich	India	Jawaharlal Nehru Technological University Hyderabad	Bachelor (Tech.) Electronics and Communication Engineering (ECE)	Bachelornote 2.8185	17	1.64	2	Daniel Reissner 2.
13	1475354 ()	Nnaemeka Edeh männlich	Nigeria and	Enugu State University of Science and Technology	Bachelor Computer Engineering (CE)	Bachelornote 2.9275	28	3.68	2	Daniel Reissner: se
15	1477768 ()	Mohammad Ashraful Alam Male	Bangladesh	Chittagong University of Engineering and Technology	Bachelor Electrical and Electronis Engineering (EEE)	Bachelornote 2.66286	10	3.24	4	Daniel Reissner:un
16	1457645 ()	Saseendhiran Srinivasan male	Indien b1 a2	Anna University Chennai	Bachelor Mechanical Engineering (ME)	Bachelornote 2.626	9	1.9	4	Daniel Reissner: or
17	1486352 ()	Md Munibul Hafiz Male	Bangladesh First Division German course	Rajshahi University of Engineering and Technology	Bachelor Electrical Telecommunication Engineering (ETE)	Bachelornote 1.43333	13	1.56	2	Daniel Reissner:
18	1480249 ()	Swapnil Ghawghawe male	Indien A1	University of Pune	Bachelor Electrical Telecommunication Engineering (ETE)	Bachelornote 2.6635	36	3.96	4	Daniel Reissner:
19	1484782 ()	Nusaeb Only Alam	Estonia	American International University-Bangladesh	Bachelor Software Engineering (SE)	Bachelornote 1.57	27	1.64	2	Daniel Reissner:

Fig 4.10: Tables with Applicant data at final stage

The above figure 4.10 shows the database tables assigned to each and every application. Once the complete processing of the application is done, Id values are generated and here we can have complete document avalibles for each and every application. In this web application, we can enable an option that can send applicant email if application is accepted.

The below figure 4.11 is from Tesseract-NetBeans IDE 7.2 Beta version. In which we implemented the code for the competing process of obtaining Tif document, calling OCR and reading data from a document and converting it to text fragments. Use these fragments to map courses to ASE study path. Once courses are mapped we send them to web page for further processing.

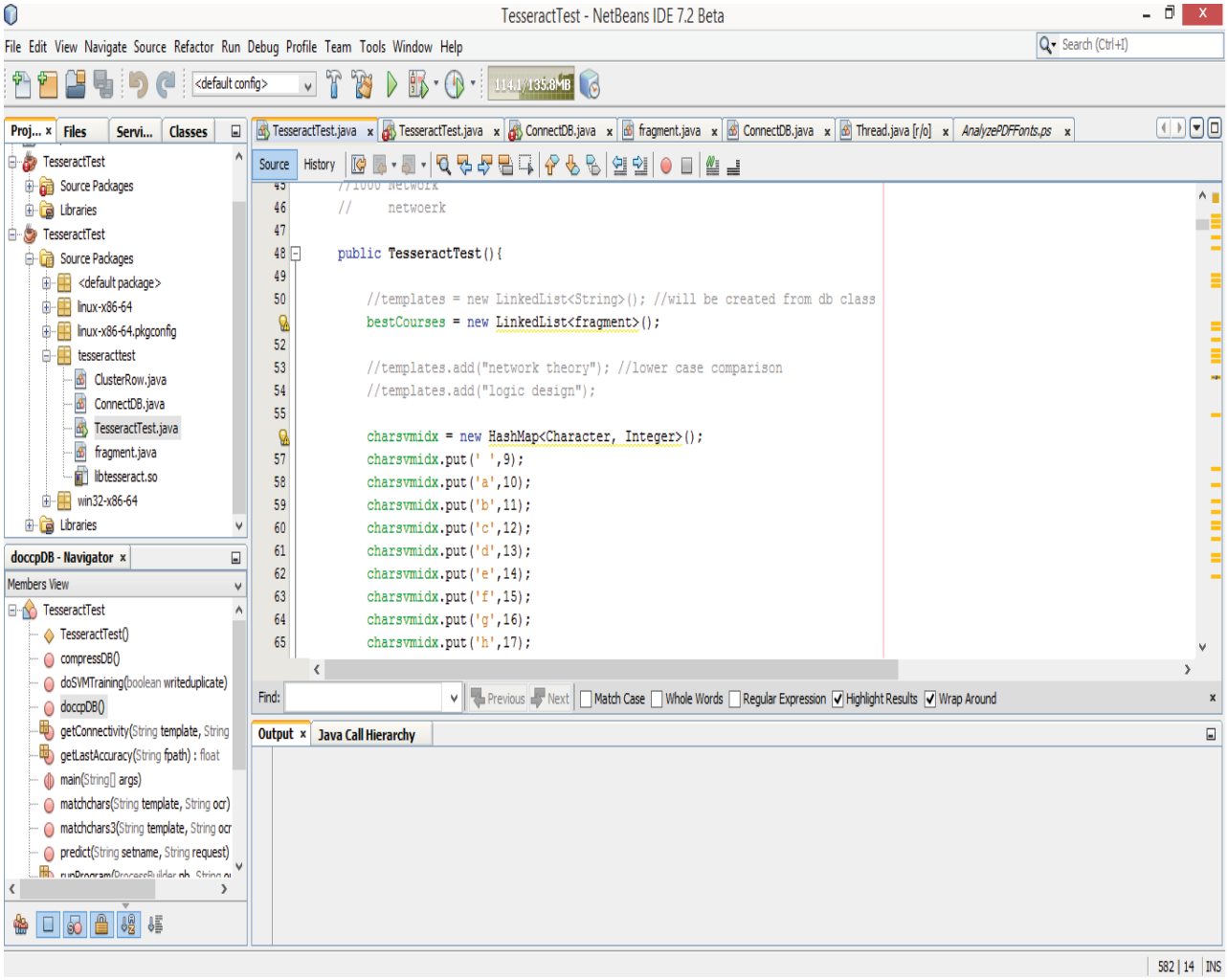


Fig 4.11: Screenshot from NetBeans

The below figure 4.12 shows the code for the training module, initial stage it generates array from character strings and assigns each character with an alphanumeric number increment of 0.25.

```
public void doSVMTraining(boolean writeduplicate){

    LinkedList<String> courselist = mysql.getASECourses();
    String svmtext = "";

    //generate svm array from strings
    //length von cours
    //for each course
    for(String elem : courselist){

        System.out.println(elem);

        Map<Character, Float> alpha;

        alpha = new HashMap<Character, Float>();


        //iterate character and assign to alpha number increment of 0.25f
        for(int i=0; i<elem.length(); i++){

            char curchar = Character.toLowerCase(elem.charAt(i));

            float oldval = 0;

            if(alpha.containsKey(curchar)){
                oldval = alpha.get(curchar);
```

Fig 4.12: Screenshot of code form Tesseract to train data

```
public boolean predict(String setname, String request){

    //1. write request as one line to file test.txt
    try (BufferedWriter out = new BufferedWriter(//buffered writer writes as expected immediately
    new FileWriter("C:\\Users\\User\\Desktop\\TesseractTest222\\TesseractTest\\libsvm\\test.txt",false))) {
        out.write(request);
    } catch (IOException e) {
        e.printStackTrace();
    }

    //2. scale to test.scale
    ProcessBuilder pb0 = new ProcessBuilder("C:\\Users\\User\\Desktop\\TesseractTest222\\TesseractTest\\libsvm\\svm-scal
                                        "-r",
                                        setname+"-train.txt.range",
                                        "test.txt");
    runProgram(pb0, "C:\\Users\\User\\Desktop\\TesseractTest222\\TesseractTest\\libsvm\\test.scale", false);

    //3. call svm predict
    ProcessBuilder pb = new ProcessBuilder("C:\\Users\\User\\Desktop\\TesseractTest222\\TesseractTest\\libsvm\\svm-predi
                                    "test.scale",
                                    setname+"-train.txt.model",
                                    "predict.txt");

    //in case of many zeros waring is written in test.scale so avoid by cleaning with second line
    //open scale file
```

Fig 4.13: Screenshot of code form Tesseract to predict courses

The below figure 4.13 shows the formulas we coded for prediction of courses. Here we used three different formulas for detecting the courses and mapping to ASE study path. We differentiated courses with length four, three and greater than four. If the course length is greater than four desired parameters like connectivity, the border is calculated and if connectivity > border is found then courses are mapped to ASE study path.

In our experience with auditing, educational certificates of different universities, we found some special course names with four and three character lengths, for example JAVA, C++ etc. These course lengths cannot be mapped with the general formula so made a different formula to map these courses. We just keep comparing each and every character and if the complete course lengths are matched, then these courses are given a bonus of 0.01 for a course with length four and 0.10 for courses with three at length. With this parameter, it can be easily mapped in character cluster preference.

```
    int b = 1;
}

int curocrlen = slinearr[j].trim().length();

if(slinearr[j].trim().length() ==4 && elem.trim().length() ==4 ){

    connecttest = tes.matchchars(elem.trim().toLowerCase(),slinearr[j].trim().toLowerCase());
}
else if(curocrlen <=3 && elem.trim().length() <=3 ){

    connecttest = tes.matchchars3(elem.trim().toLowerCase(),slinearr[j].trim().toLowerCase());
}
else{

    connecttest = tes.getConnectivity(elem.trim().toLowerCase(),slinearr[j].trim().toLowerCase());
    //problem software weggeschnitte wie drawing sollte aber negativ beachtet werden
}

//4. compare for best connectifity and bigger 0.7
if(connecttest>0.75 && connecttest>maxconnectivity){

    maxconnectivity = connecttest;
    bestCours = elem;
    besttyp = felem.typ;
```

Fig 4.14: Screenshot to formulate 4 and 3 lengths courses

Chapter 5
Results

In chapter we see the results obtained from complete thesis work and discuss more in detail about them. The below screenshot is obtained after code is debugged and run. This shows, training the data by LIBSVM and accuracy of data is 100% from which SVM classification is done and this is also used in a cluster.

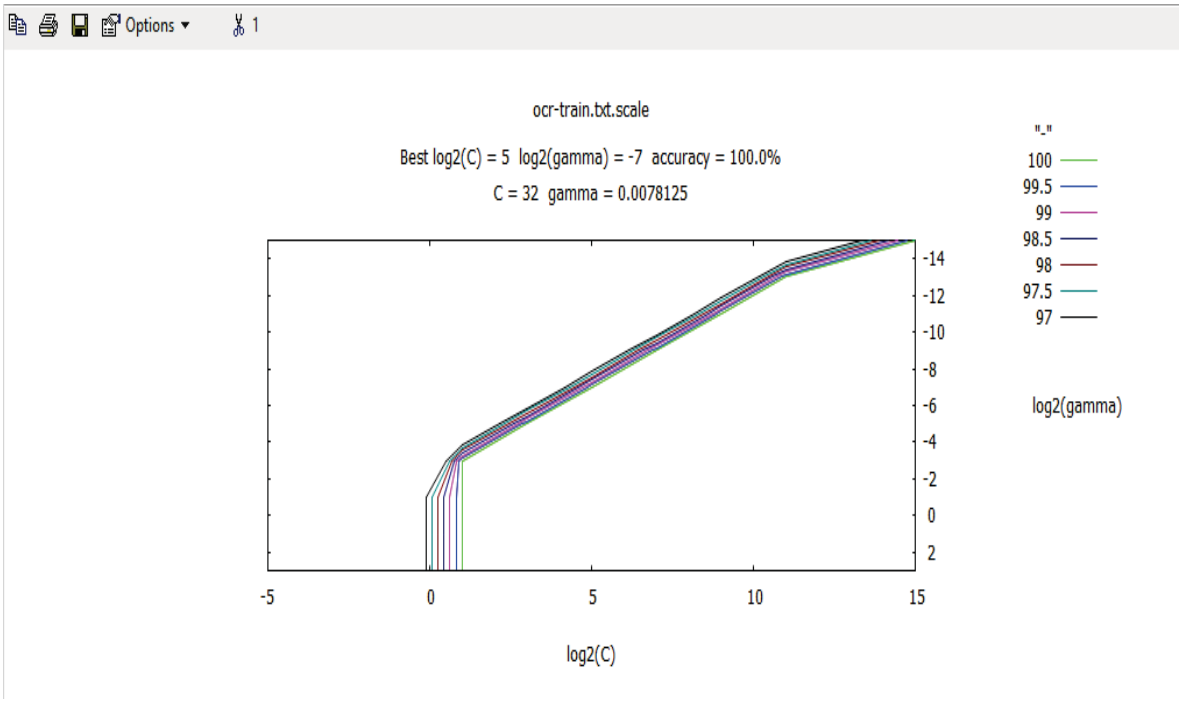


Fig 5.1: Training data from LIBSVM

Below figure 5.2 shows first result from Tesseract, once the data is trained the pre-filled data in the database are called and this data will be compared with the text fragments from OCR. This starts with university id followed by the ASE related course and for these courses Character Cluster Preferences are generated.

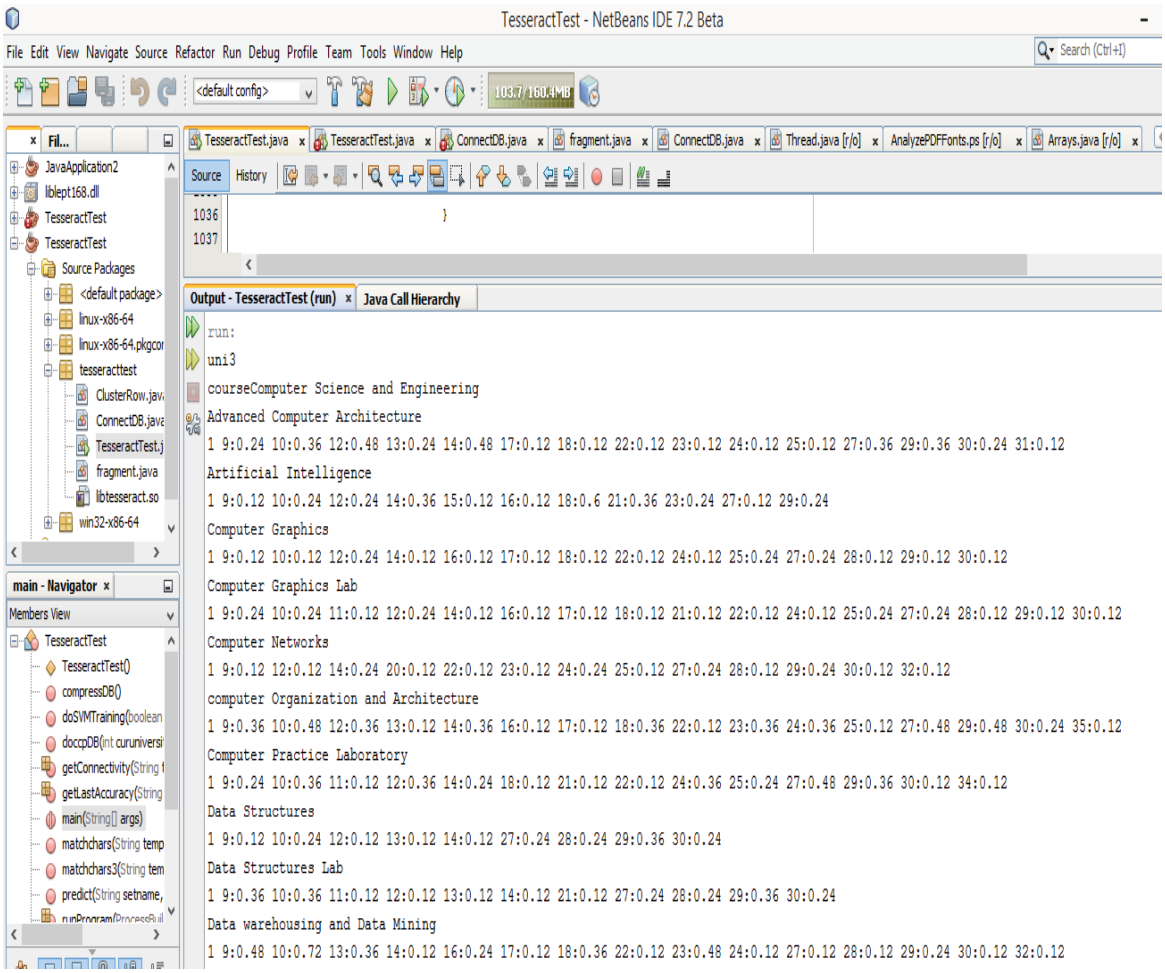


Fig 5.2: Generating Character Cluster Preferences

Uni 3 in above figure shows the university id from DB tables with course name as Computer science and Engineering. A further character string is created for each and every course related to ASE study path.

The figure 5.3 shows insertion of courses into character cluster preference after detection of the university, all the courses that are listed as ASE preferred courses are initialized. A character string is created for further classification and mapping of courses to ASE study path.

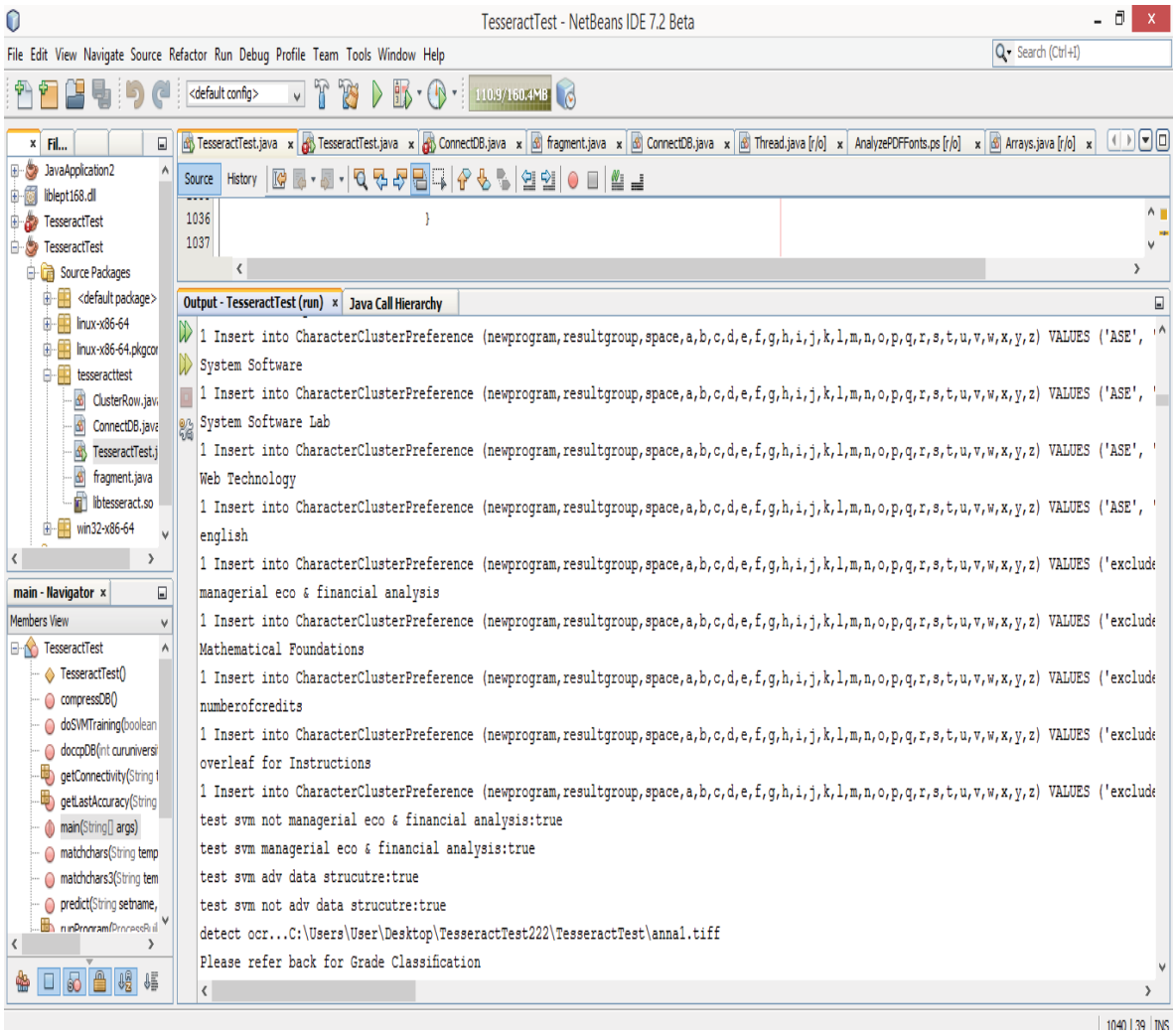


Fig 5.3: Initializing DB course list

Below figure 5.4 shows the OCR detection of all courses from certificate. This detection works differently for each and every university. As the text and numbering format changes from university to university, we need to code them in different ways to generate these data. This data is further used by Tesseract to work with SVM and clustering methods to map courses.

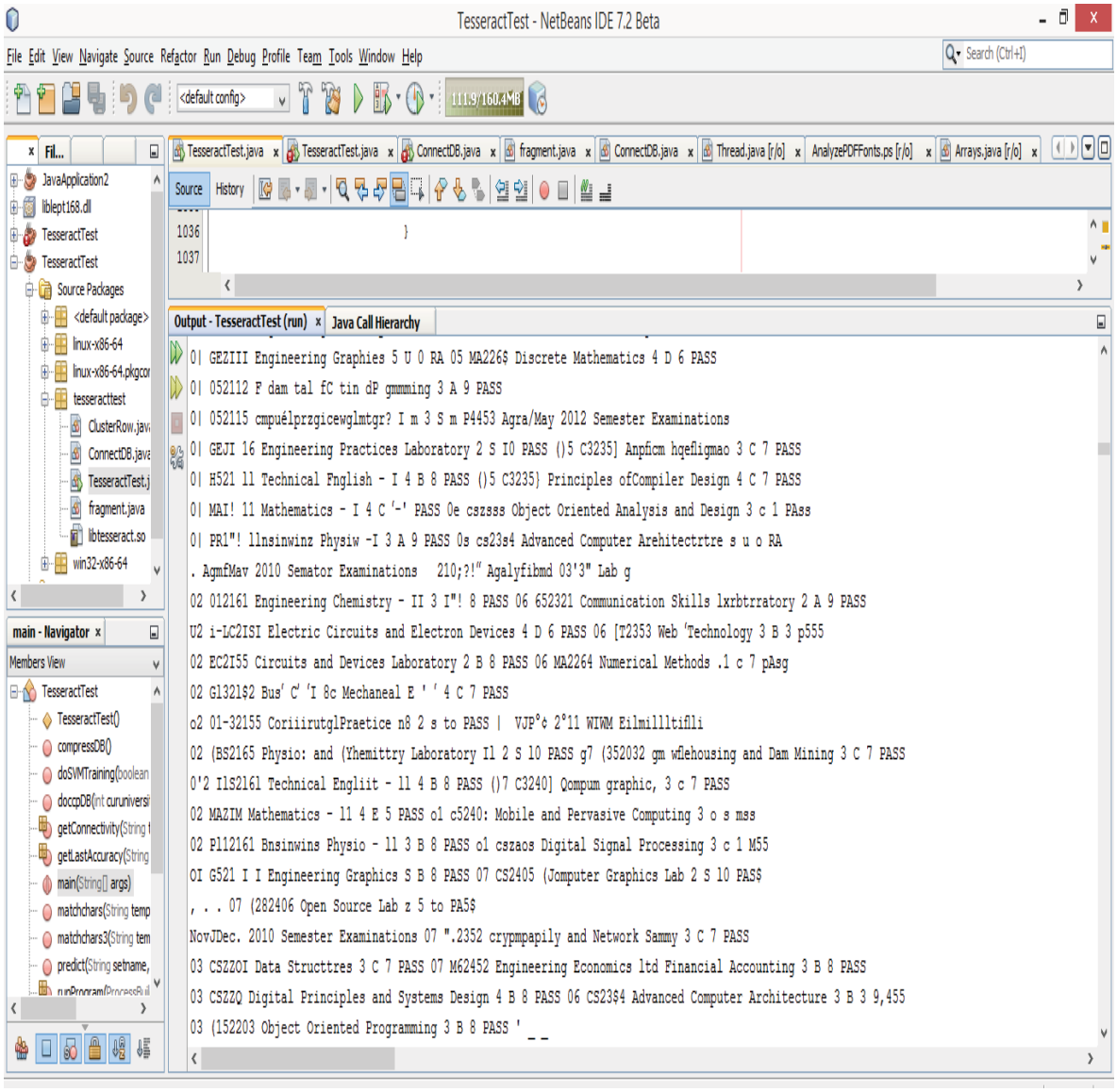


Fig 5.4: Java tool output of detecting OCR text

The figure 5.5 shows the results while calculating parameter values like boarder, distance and connectivity. With this obtained value mapping of courses is done on further stage. Here we can observe the result of both SVM classification showing if is related to ASE or not and also cluster result showing if it is selected or not for the ASE study path.

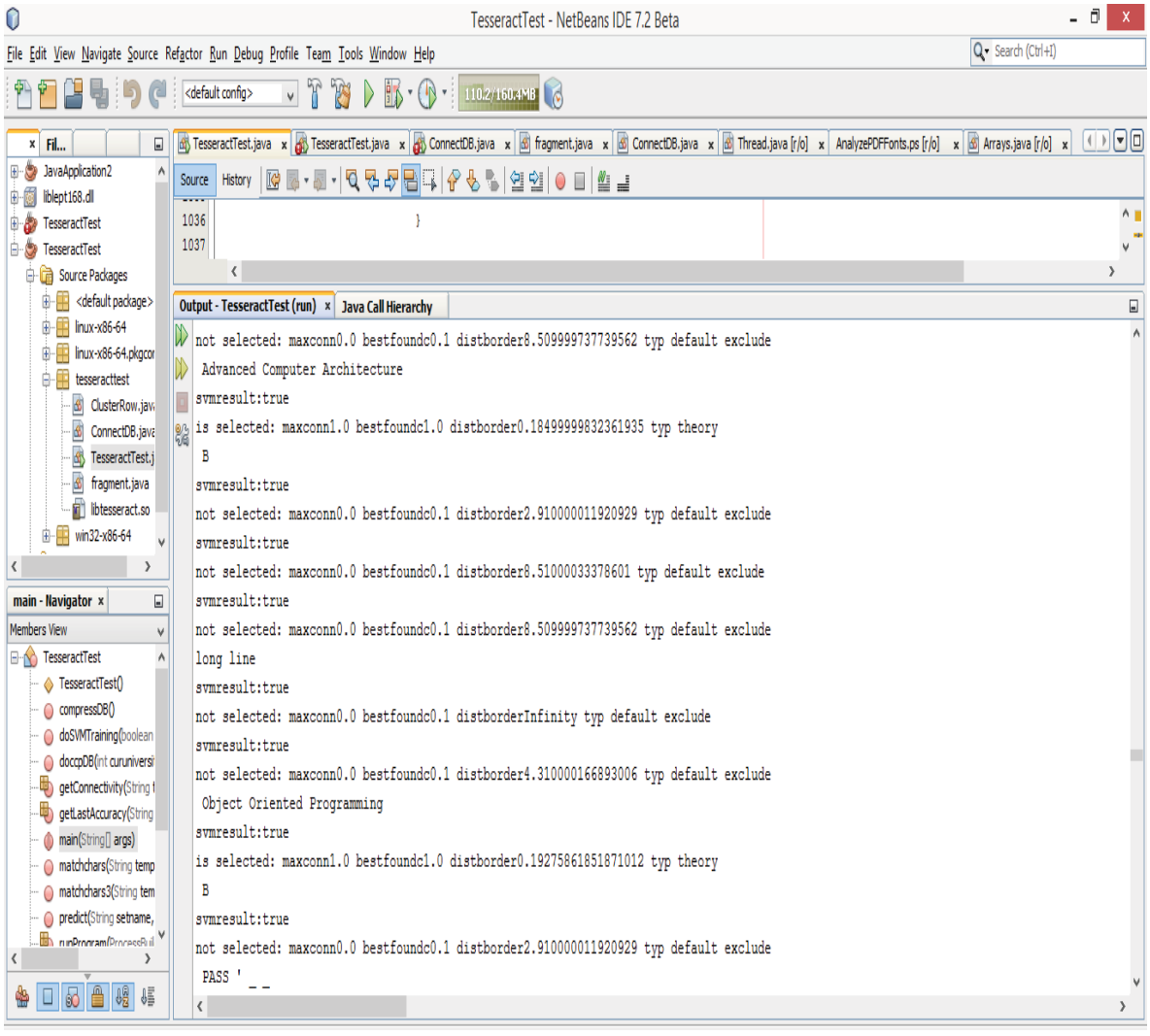


Fig 5.5: output of SVM and cluster classification

The figure 5.6 shows the final results after mapping best courses to ASE study path. Here we can find a course name with obtaining marks as output. The firs course name of every line is from the DB filled data followed by OCR detected sting. This data can be further moved to web application where we can set a formula to obtain some value for all this ASE related courses.

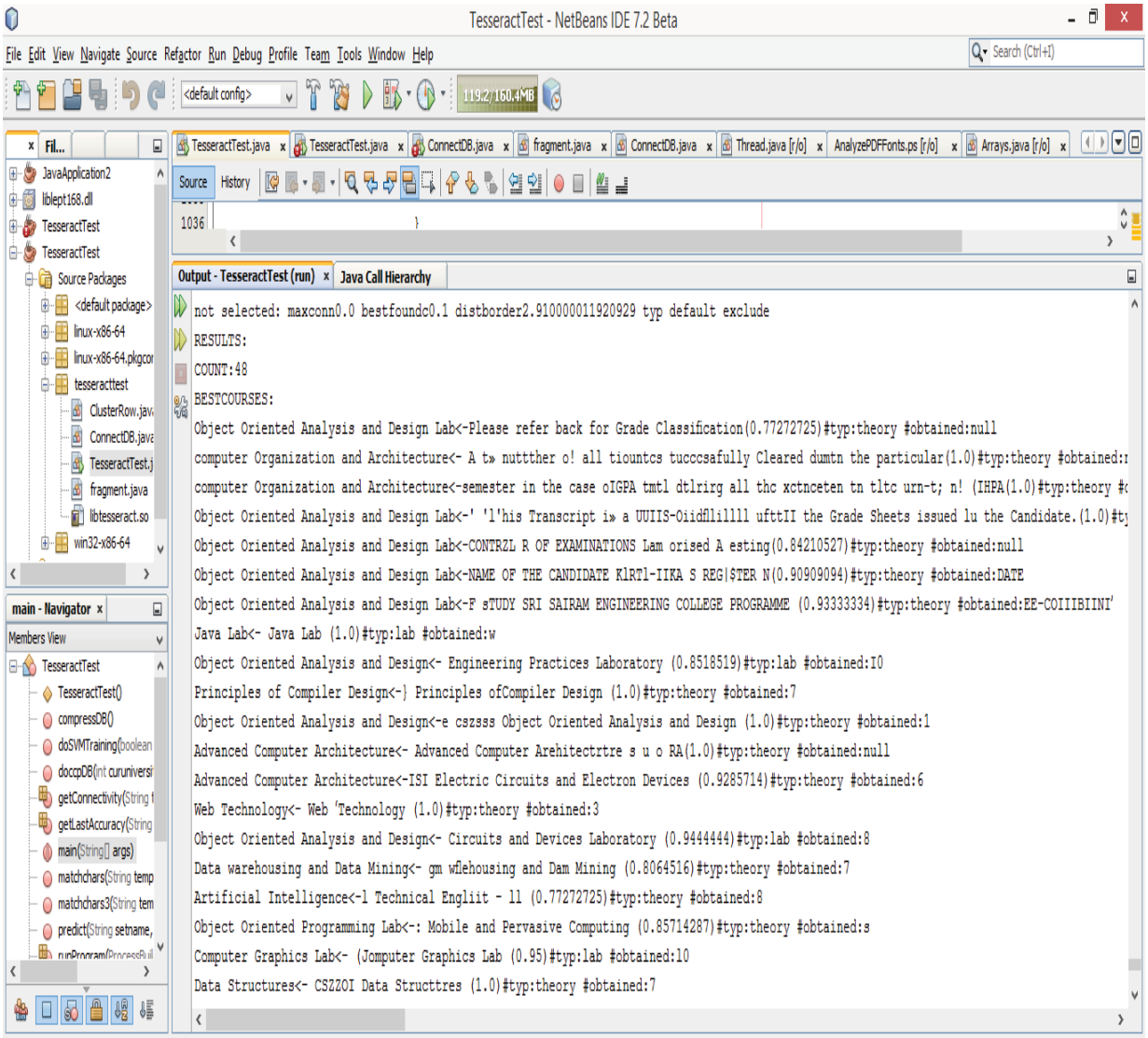


Fig 5.6: ASE mapped courses

We have implemented this work on two universities JNTU Kakinada and Hyderabad course modules. The results obtained are very much convincing and algorithms and formulas implemented in code worked very well. With this work I can confidently say that mapping the courses of any university to ASE related courses can be done by making some slight changes in split and merge condition. The below shown tables give you a clear data analysis on implementations.

JNTUK	
Manual Passed input	59
Manual ASE Courses	36
Manual Not in ASE	23
OCR Mapping	59
OCR ASE Mapping	36
OCR Not ASE Mapping	23
Tesseract split and merge	50
Tesseract Detected ASE Courses	36
Tesseract Detect not ASE	9
University document excludes strings	5
Cluster mapping	59
Cluster ASE	36
Cluster not ASE	23
SVM Mapping	59
SVM ASE	41
SVM not ASE	6
SVM noise	310
SVM noise assigned to ASE	243
Not ASE courses assigned to ASE	23

Table1: Data analysis of JNTUK

For the university JNTUK total number of courses studied by student are 59 and in this course, there are 36 courses which are related to ASE study module and 23 not ASE related courses. At the initial stage of OCR detection all these courses are given as input in tiff format. OCR is successful in detecting all these courses and there are some other text strings that are read by OCR. Next comes the split and merge stage to map the courses to an ASE study path made by Tesseract.

With this split and merge formulas implementation our work in mapping courses to ASE study path is fulfilled and other courses not related to ASE are filtered. There are other text strings from file that are detected and mapped as to exclude case. Classification of text strings obtained from OCR is done by SVM which says if it is related to ASE or not. And the mapping of courses to right course is done by clustering. From the table 1 it is very clear that SVM adds some noise stings in processing and this leads to decrease in its efficiency, whereas cluster works fine in mapping courses and its pro from noise.

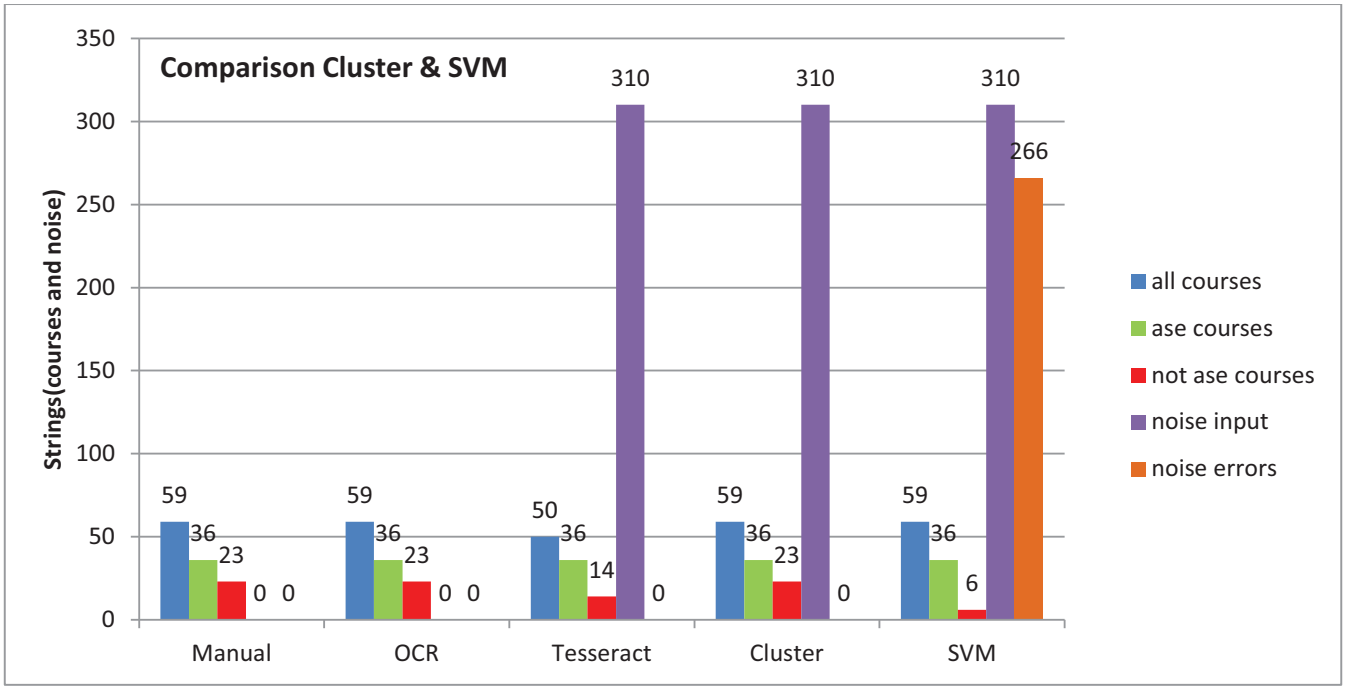
The other parameter which we are very much concerned is about the detecting obtained marks from applicant input file. There are many parameters that need to be considered in detecting obtained marks, as the row and columns differ from university to university, we made formulas in split and merge to obtain these values. Below table2 shows the detecting rate of obtained marks.

Detected ASE Marks	13
Fail	23
Not ASE detected	2
Not ASE fails	10

Table2 : Detection of obtained marks

In case of JNTUK we see the file we give as input is not in good clarity so OCR and Tesseact count not work that efficiently in detecting obtained marks but we see a linear change in the results of second university. There were detection in both ASE related courses and not ASE related courses that are separately mentioned in table2.

A graph is plotted by the above observation of table1. In this graph we can have a detail markings of courses at different phase like manual input we provide, OCR detection, Tesseract split and merge state and classification with SVM and Cluster mappings. The last bar shows the noise that is included in the process and we see this is filtered by clustering process but could not be filtered by SVM.



Graph1: comparison of Cluster and SVM in all phases

With similar formula and slight change in formulas, mapping is done for second university and data analysis and graphs plotted are shown below.

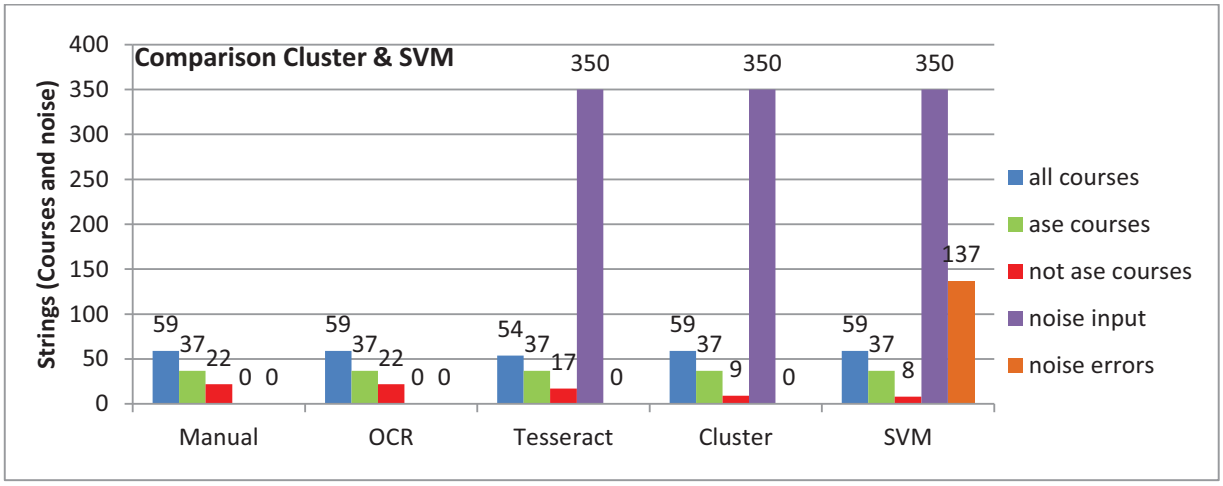
JNTUH	
Manual passed input	59
Manual ASE courses	37
Manual not ASE	22
Tesseract split and merge	54
Tesseract Detected ASE Courses	37
Tesseract Detect not ASE	9
University document excludes strings	8
Cluster mapping	59
Cluster ASE	37
Cluster not ASE	9
SVM Mapping	59
SVM ASE	37
SVM not ASE	8
SVM noise	350
SVM noise assigned to ASE	133
Not ASE courses assigned to ASE	5

Table 3: Data analysis of JNTUK

Below table 4 shows detection of marked obtained. Formulas and algorithm remains same for both universities for this process, but we observed a good detection rate when compared with the first university. The mail reason behind this would be clarity of the input tiff file we provided.

Detected ASE Marks	35
Fail	2
Not ASE detected	8
Not ASE fails	1

Table4: Detection of obtained marks



Graph2: comparison of Cluster and SVM in all phases

The below graph shows the result from SVM and Cluster based approaches. The graph is plotted for correctness of word strings by both approaches. We see more correctness in cluster based approach when compared to SVM.

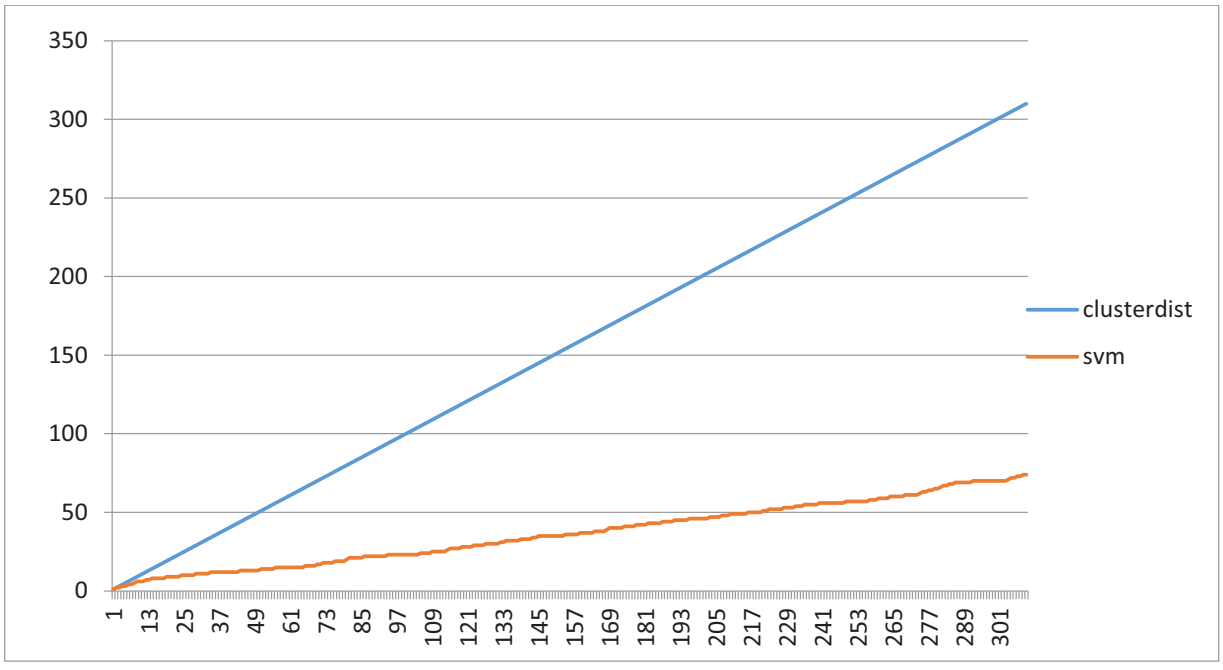


Fig 5.7: Graph with SVM and cluster correctness

Chapter 6

Future Scope

This thesis work is a first project that is executed on an idea to create a tool to optimize pre-processing stage in the application process at TU Chemnitz. With this work I can say we came to conclusion that if we extract ASE related courses from the applicant certificate, we can save much more processing time and we can assure the subjects that are considered are most perfect and auditing of application is very productive. After working on this project I see there is much more scope for improving the tool and many new features can be added to it. In this chapter I want to discuss phase vise developments that can be made.

The first phase will be OCR detection, for more accurate results from SVM and Clustering we need to put much more effort into OCR working. There is one thesis project engaged in Deartment of Computer Science on OCR, so if we see much better results from that thesis work we can add it to our project such that results from SVM and Cluster can be improved further. In out thesis work I played with the image clarity and improvements in brightness and contrast for better detection of characters from the scanned copy of applicant certificate. It was trial and error policy I followed to fix the pixel size and other parameters. So if we see any good results from OCR detecting normal images or files and extract data than that would help a lot to improve working of this tool.

The second phase of improvement is from Tesseract, I wrote Java code so it is very flexible that we can change the code as per our requirements. Improvements I can say from Tesseract are from handling SVM and clustering parts in it. There are different algorithms that are implemented to extract courses with a very less number of characters to many numbers so, if possible, if we make

good improvements in it we can see better results in mapping the courses. As I was completely concerned about the mapping of courses hope I am successful in it, but there are some other factors like extracting the marks of each and every course that is extracted which will be a big task ahead. There should be some major developments in Tesseract to extract numbers and map it to ASE list. The code for each and every universities changes to some extent as the format of the exam sheet changes from university to university. The sheet sometimes has a single box with course name and marks awarded and for some university, we observe two sets of boxes vertically with course name and marks obtained. With my work experience I see there are different formats in marks listing in certificates as some university gives the minimum and maximum marks and then obtained marks. And in some university certificate we see obtained marks directly into different grading systems. So when we are planning to extract marks from these universities, we need to implement the different code structure to fetch course name and obtained marks. There is a further task to calculate this obtained marks and apply some scaling factor to make it more generalized for all the universities.

The third phase will be in the database. Now we implemented this work with three or four numbers of universities which can be increased to many number of universities. Once the database is set up with huge data, we can add many improvements and new features to this tool. With this thesis work inital database design and tables for further work is completely laid out. As mentioned in chapter 4 with the figure number 4.5 SVM database setup is initiated such that after obtaining much data from its many improvements in SVM can be performed. Much more optimisation is observed after collaborating may web applications that we used to pre-fill the database. There is much more that focus needs to be placed in extracting ASE related courses from many other universities which are often with many applications. All this data need to be pre-filled in our DB.

The fourth phase of development is from GUI. The web applications we used are for testing initial startup of the project, so once the final tool is setup, then we need much more well designed web applications with high end database support. There are some separate application that can be developed for users or applicants and for auditors or administrators.

There is much more scope for new implementation in this tool, one best idea which I got after working very closely with SVM and clustering modules is to implement SVM in classifying the students with their bachelor studies with a particular study path like Electronics and Communication Engineering, Computer Science Engineering, Information Technology, Machnical Engineering, etc. come into ASE with a percentage of marks and after working hard they pass out successfully with good grading then we can try classifying the students from a particular study path in bachelors with the percentage of marks obtained in Masters course such that we can suggest the one who is coming new to masters course can get an idea with what percentage of marks he can be out from masters. This is with a condition on how good he can work hard. If this is developed with the all other master courses we can observe very peculiar results that would be more intresting to find and analyse.

A. Contents on CD

TesseractTest1	Here is complete code and supporting files of the code implementation. This is the folder with one example implementation of the JNTUH University. Input files which are provided to OCR will present in this folder.
TesseractTest1/libsvm	This folder has code that can work with LIBSVM implementation. All other supporting files for SVM implementation on obtained data will be in this folder.
TesseractTest1/Tess4j	In this folder we have contents of code related to Tesseract working.
TesseractTest1/log.txt	In this file we can see the classification vales written by SVM and Tesseract. If there is any error in written code we can cross check it from here.
TesseractTestSVM	Here is complete code and supporting files of the code implementation. This is the folder with one example implementation of the JNTUK University. Input files which are provided to OCR will present in this folder. All other sub folders described above will be repeated here.
Results Folder	Here I provide data analysis sheets which I have worked to obtained the graphs shown in results.
Document Folder	Complete documentation is saved in this folder

Bibliography

[1] Ivan Dervisevic
Machine Learning Methods for Optical Character Recognition December18, 2006

[2] Ray Smith
An Overview of the Tesseract OCR Engine, Google Inc.

[3] Jiawei Han, Micheline Kamber, Jian Pei
Data mining Concepts and Techniques, Waltham, MA organ aufmann/Elsevier, c2012

[4] Data Mining courses:
http://web.fhnw.ch/personenseiten/taoufik.nouri/Data%20Mining/Course/Course5/DM-Part5.htm (last accessed on 20.6.2015)

[5] Data Mining: http://docs.oracle.com/html/B14339_01/4descriptive.htm (last accessed on 20.6.2015)

[6] Clustering: http://marktab.net/datamining/2010/07/24/microsoft-clustering/ (last accessed on 25.6.2015)

[7] Vasanth Nemala
Efficient clustering techniques for managing large datasets, University of Nevada Las Vegas

[8] Ajiboye Adeleke R,Isah-Kebbe Hauwau, Oladele Tinuke O.
Cluster Analysis of Data Points using partitioning and Probabilistic Model-based Algorithms

[9] Ryan Rifkin
Multiclass Classification 9.520 Class 06, 25 Feb 2008

[10] Achmad Widodo
Support vector machine in machine condition monitoring and fault diagnosis, Mechanical Systems and Signal Processing 21 (2007) 2560,Äi2574

[11] D.Michie, D.J. Spiegelhater, C.C.Taylor
Machine Learning, Neural and statistical Classification, C.C.Taylor Feburary17, 1994 Research Internship report

[12] Abdul Sami, Kondreddy Mahendra, Abhishek Diddikadi
Research Internship Report

[13] Julian Vitay
AI lab, Dept. of computer science, Tu Chemnitz @bookletSVM-Machine learning

[14] SVM Classification parameters: http://php.net/manual/en/svmmodel.predict.php (last accessed on 20.6.2015)

[16] Ning Li
An Implementation of OCR system Based on Skeleton Matching. Computing Laboratory, University of kent at Canterbury, United Kingdom August 1991

[17] Chih-Wei Hsu, Chih-Chung and Chic-Jen Lin
A Practical Guide to Support Vector Classification, Department of Computer Science, National Taiwan University, Taipei 106, Taiwan. http://ntucsu.csie.ntu.edu.tw/~cjlin/

[18] Patrice Y. Simard, Richard Szeliski, Josh Benaloh, Julien Couvreur, and Iulian Calinov
Using Character Recognition and Segmentation to Tell Computer from Humans, One Microsoft way, Redmond, WA 98052.

[19] Sushruth Shastry, Gunasheela G, Thejus Dutt, Vinay D S and Sudhir Rao Rupanagudi
“ i ” - A novel algorithm for Optical Character Recognition (OCR), Worldserve Education, Bangalore India.

[20] Mehdi Salehpour, and Alireza Behrad
Cluster Based Weighted SVM for the Recognition of Farsi Handwritten Digits. 978-1-4244-8820-9/10/$26.00 ©2010 IEEE