MATHEMATICAL FORMULA DETECTION IN TECHNICAL DOCUMENTS

INTRODUCTION

Mathematical formulae are common components in many scientific documents, and their automatic recognition, in order to extract structural and semantic information, has been researched from as early as 1968 [1]. Nevertheless, mathematical formula recognition remains a difficult research problem as the lack of predefined structures of formulae, together with the large number of non-standard symbols and fonts, prevents their easy extraction, recognition and structural analysis.
Up to now, the majority of formula recognition methods target images or rasterised documents. However, due to the popularity of PDF for electronic publishing and sharing of scientific documents, a new and important field of document analysis is emerging, which is the direct analysis of PDF files.
Compared with images, PDF documents can provide richer information such as Unicode, fonts and coordinates, which can be mined to improve and complement the traditional document analysis techniques adopted for images. In this paper, we will focus on one of the most essential steps of formula recognition in PDF documents: determining where formulae are located by detecting their precise boundaries,
namely formula identification.

Mathematical formula identification from documents has been researched for more than twenty years, but until now, the performances of these methods have not been at a sufficient level to be adopted in realistic application scenarios:
The main obstacles can be summarised as follows:
1. Most existing methods target image documents, which heavily rely on mathematical symbol recognition. Considering that math symbols in images are not yet satisfactorily recognised, errors produced by OCR inevitably propagate throughout formula identification, heavily reducing its performance.
2. Most of the existing formula identification methods are purely rule-based. Although they work well for limited sets of documents with simple layouts, a common shortcoming of the rule-based methods is parameter

setting.
To our best knowledge, parameters in purely rule-based formula identification methods are mostly set by experience.
3. To overcome the problems of rule-based methods, several machine learning techniques are introduced. Although these learning-based algorithms can solve some of the existing problems, they are still at the preliminary study stage.
4. Due to private datasets and closed-source code, it is almost impossible to reimplement the algorithms proposed in existing papers with only limited implementation details.

According to the features used, the existing formula identification methods can be classified into two categories: character-based and layout-based. Furthermore, according to the techniques adopted, they can be classified as follows:
rule-based or machine learning-based.
The rule-based methods for mathematical formula identification detect mathematical formulae through constructing heuristic rules on different types of features, and most of the character-based methods are rule-based. Usually, they first identify special math symbols (e.g., “=”, “+”, “<”), which only appear in mathematical formulae rather than ordinary text, then apply specific context propagation rules on these symbols according to their operator domains [10–13]. Kacem et al. [14] constructed a fuzzy logic model to identify math symbols and then utilised features of math symbols (bounding box, relationship between symbols, etc.) to merge or expand their regions to form the embedded formula area. In [15], all characters were first recognised by traditional OCR engine and then the outliers were considered as the candidates of mathematical symbols. Along this direction, Suzuki et al. added verification rules according to positions and sizes of characters in order to develop a dedicated OCR system for mathematics documents. To identify isolated formulae from PDF, Baker et al. established rules according to the proportion of plain text words in a line to discriminate isolated formulae from ordinary text lines.
Based on the assumption that formulae are usually typeset with different layout features, such as isolated formulae being larger and more sparse than plain text lines, many layout-based methods to identify formulae have been proposed . Chowdhury et al. [19] proposed a recognition-free method and built decision trees based on layout features to distinguish formula lines from plain text lines. Another recognition-free method [20] extracted

mathematical expressions using image segmentation technique. Although this approach only relied on projection features, the segmentation thresholds in this approach were hard to set, especially for unknown types of documents. Besides, it did not work for embedded formula extraction. Garain al. identified isolated formulae through comparing features of a text line with features of the averages of all text lines in a document. To extract embedded formulae, they first adopted n-gram models to identify text lines, which might contain embedded formulae and then constructed rules based on common typographical conventions of mathematical expressions.
One common problem with rule-based methods is that they inevitably introduce decision parameters of predefined rules. However, it is difficult to set optimal values for these parameters manually or empirically. In addition, these parameters are sensitive to different formula types, document layouts, styles and fonts.
To solve the problems in rule-based methods, machine learning-based methods are proposed. The main idea of several isolated formula identification methods is to consider each text line as an instance and build classifiers to decide whether a text line is a formula or not. The main difference between those learning-based methods is the choice of features and learning algorithms. For example, Jin et al. exploited Parzen Windows technique to identify isolated formulae. Drake et al. adopted computational geometry features of the neighbour graph of connected components, which are the results of the Voronoi Graph Analysis. Based on the assumption that complex page components (e.g., table, formulae, graphs) are more sparse than plain text lines, Liu et al. exploited SVM (support vector machine) and CRF (conditional random field) to identify sparse lines, then applied rules to distinguish formulae from other complex page components (e.g., tables and graphs) among the sparse lines.
Although the preliminary progress of adopting machine learning techniques in formula identification has been made, several problems still exist in current machine learning-based methods: 1) Features utilised to build the classification models are not discriminative and informative enough to adapt to various types of formulae and documents. Although simplistic features may work well in certain documents, it may cause underfitting problems and fail to deal with large and diverse documents. 2) Several learning algorithms are attempted in this area and proven to outperform the rule- based methods.

APPROACHES FOLLOWED:

1)Crude Method:

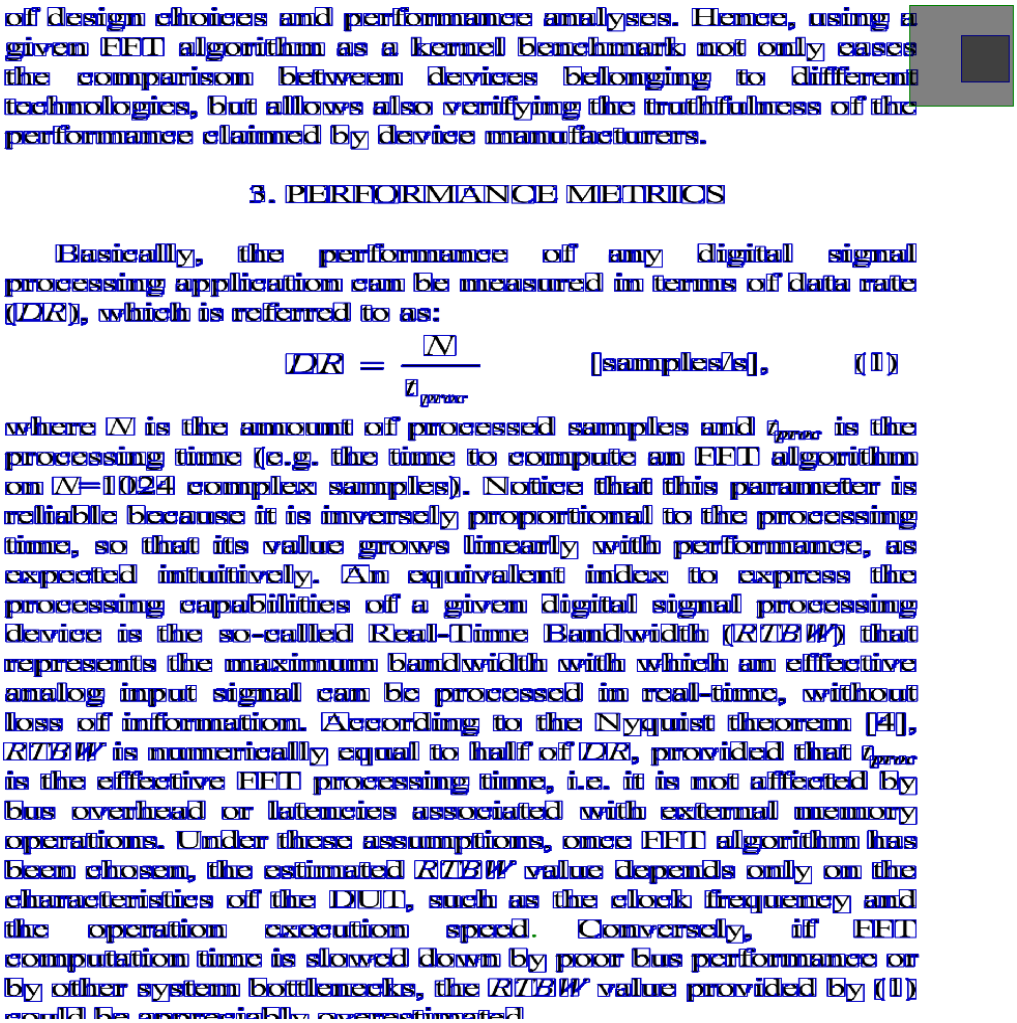
This was a method I followed initially.The proposed basic flow is goven below:

-Initial processing or pre-processing of documents.

-Converting the image to gray-scale.

-Finding contours of the gray scale image.

-Defining and constructing a bounding-box or rectangle around each contour(character in our case).



of design choices and performance analyses. Hence, using a given FFT algorithm as a kernel benchmark not only eases the comparison between devices belonging to different technologies, but allows also verifying the truthfulness of the performance claimed by device manufacturers.

3. PERFORMANCE METRICS

Basically, the performance of any digital signal processing application can be measured in terms of data rate (*DR*), which is referred to as:

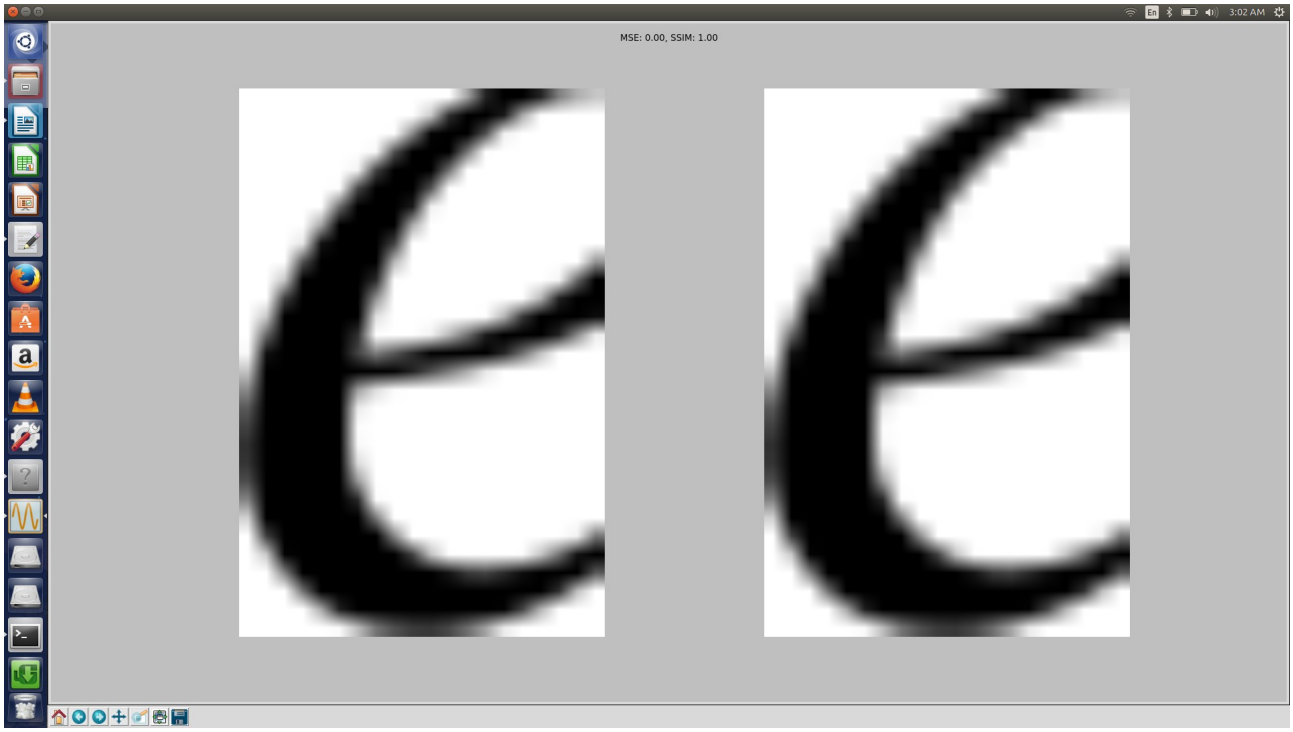
$$DR = \frac{N}{t_{proc}} \quad [\text{samples/s}], \qquad (1)$$

where N is the amount of processed samples and t_{proc} is the processing time (e.g. the time to compute an FFT algorithm on N=1024 complex samples). Notice that this parameter is reliable because it is inversely proportional to the processing time, so that its value grows linearly with performance, as expected intuitively. An equivalent index to express the processing capabilities of a given digital signal processing device is the so-called Real-Time Bandwidth (*RTBW*) that represents the maximum bandwidth with which an effective analog input signal can be processed in real-time, without loss of information. According to the Nyquist theorem [4], *RTBW* is numerically equal to half of *DR*, provided that t_{proc} is the effective FFT processing time, i.e. it is not affected by bus overhead or latencies associated with external memory operations. Under these assumptions, once FFT algorithm has been chosen, the estimated *RTBW* value depends only on the characteristics of the DUT, such as the clock frequency and the operation execution speed. Conversely, if FFT computation time is slowed down by poor bus performance or by other system bottlenecks, the *RTBW* value provided by (1) would be appreciably overestimated.

Bounding Box Result

Hence each character in the document is separated or bound by a bounding box.

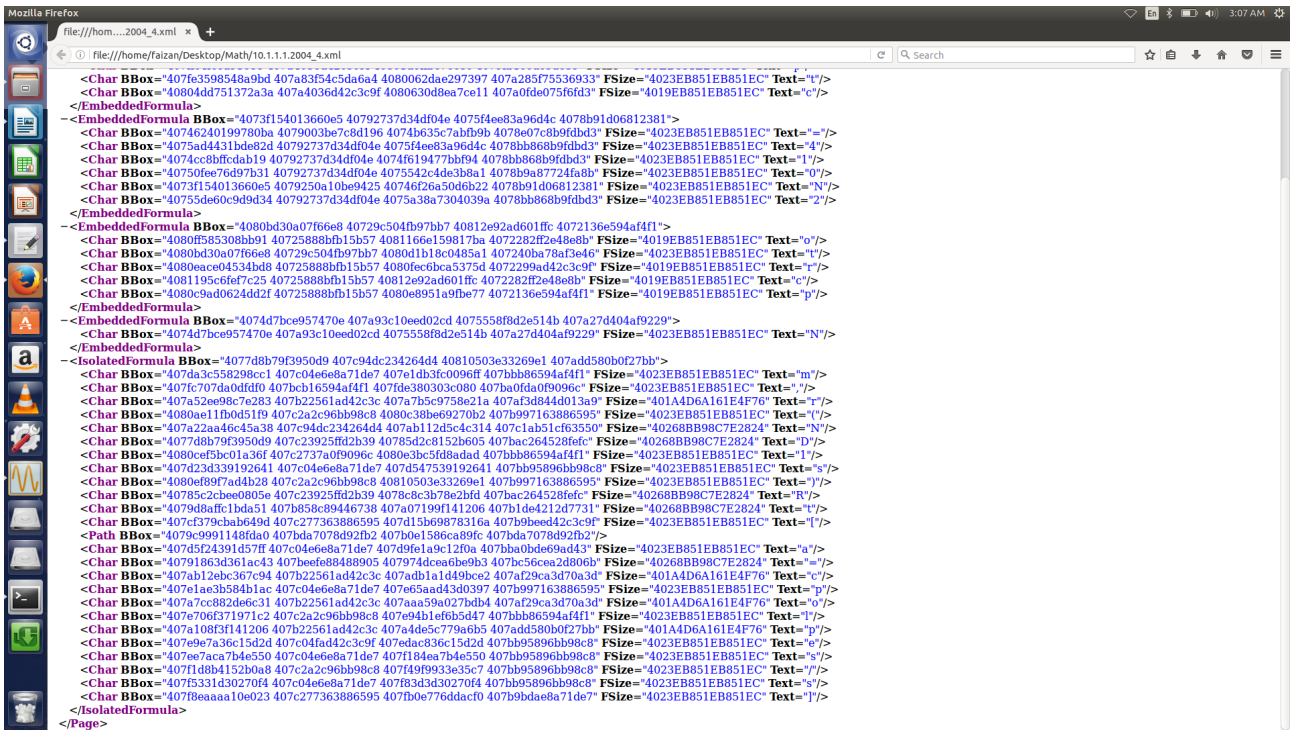
-Compare two characters using Mean squared error and structural simislarity.



Comparison between two characters

Hence classify all characters into normal or special characters.
-Hence identify the mathematical formula.

2)Method based on the paper:
The method I tried according to the paper:
-Preprocessing of the image.
-Applying Optical Character Recognition(OCR) on the given image.
-Store isolated and embedded formulae in XML format.



XML format

-Extract features like Variance in height,font size,distance between two lines and few other features.
-Train these features using various machine learning algorithms.
-Try using Boltzmann Neural networks for better accuracy.