**TITLE**

**Abstract**

This paper presents equation detection in heterogeneous document images that may contain figures, tables, text, headings, equations and then classifying them into two categories – *chemical* and *non-chemical*. In our proposed method we extracted the equations using morphological operators and histogram analysis and classified them using an open source OCR engine. The effectiveness of the proposed method is demonstrated by testing it on 89 document images in both Bangla and English language. Test results show an accuracy of 99% and 97.5% for displayed equation extraction and classification into two categories respectively.

**Introduction**

Large amount of documents are being digitized today for the purpose of storing and analysis. A lot of work has been done to detect and extract the mathematical equations present in heterogeneous document images over the past decade. Some work use the embedded file structure in pdf to achieve accurate detection result [2.1]. But recent works have surpassed the pdf limitation and started identifying math formulas in scanned document images. A method described in [1.3] does the same but it takes into account that the expressions are available in isolated form. The statistical approach taken by Garain [1.12] on the corpus of 400 pages to differentiate normal text lines and lines containing equations/expressions on the basis of their white spacing will work in chemical equations too. Jin et al [2.5] proposed a similar theory to extract displayed formulas detected by Parzen classifier constructed based on line height and indent features. Chowdhury et al [2.6] worked on segmentation on math zones by decision trees which was based on appearance on superscript and subscript and heights of math symbols. Drake and Baird [2.7] came up with a graphical approach; similarly Guo et al [2.8] developed a Gaussian mixture model to describe spatial relationship between sub-components of a math expression. Tian et al [2.9] researched layout differences between math formulas typed by Latex and Microsoft Word. Another method to check text style (regular, italic, bold) at the character level has been approached in [1.13]. The proposed method of Wei-Tan Chu [2] involves detection of features based on centroid fluctuation information of non-homogeneous regions. Kacem et al [1.8] extracted the equations using fuzzy logic by detecting math operators such as +, -, (, [ but these are common in chemical equations as well. Garain too [1] proposed a bunch of features to identify equations but they all lack the ability to segment between mathematical and non-mathematical equation. If some document contains both chemical and mathematical equations, their methods would not be able to extract only mathematical equations. More importantly, recent development in the field of chemo-informatics requires precise identification of chemical equations amongst myriad collection of chemical and non-chemical formulas/equations. This can be important for various tasks like bond-electron matrix construction, etc.

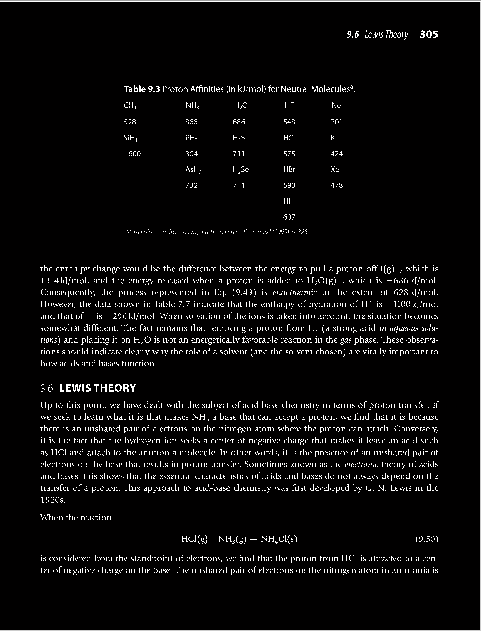
The study embodied in this paper is motivated by the aforementioned needs. The contribution of the work is threefold – *First*, text lines are segmented by taking horizontal projection and word blobs are formed using morphological operators. *Second*, the word blobs are analyzed to identify the operators (+, -, X, ∑, →, ∏) which in turn helps to identify both displayed and embedded equations using operator and operand count. *Third*, the displayed equations are run through a commercially available open source OCR (Google Tesseract 3.02) to get the textual form and identify the presence of chemical elements by matching regular expression with dictionary made out of periodic table. If the matching percentage exceeds a certain threshold β, it is categorized as a chemical equation, else as a non-chemical equation.

**Proposed method**

1. **Tall and Wide Component Removal :**

We first binarize the document image by setting intensity values of pixels in homogenous blocks as 0 and setting that in non-homogeneous blocks as 1.

The binarized document images may contain charts, tables, figures etc which will need removal as we are considering only displayed and embedded equations in this method. Our observation of document images confirms that the charts, tables, figures contain characters with height, width much more than the median height and width of all characters present in the document. So, median height and width of every character is measured and the characters containing height more than 5 times of median height and width more than 20 times of median width are removed from the original binarized document image.

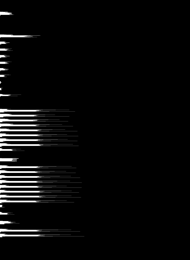


1. **Text Line Segmentation:**

To detect equations, text lines have to be segmented first. Intensity values of pixels are projected in the horizontal direction to construct a horizontal profile, and the rows having profile value lesser than a threshold value α will be removed. This is done to remove any stray components between lines. This profile is converted into a binary image having the same size as the document image and n columns corresponding to each row is set to 1 from the left where n is the value of the profile value of that row from the horizontal projection.

Now each connected component of this image corresponds to each line and we get the start and end row values for each line.

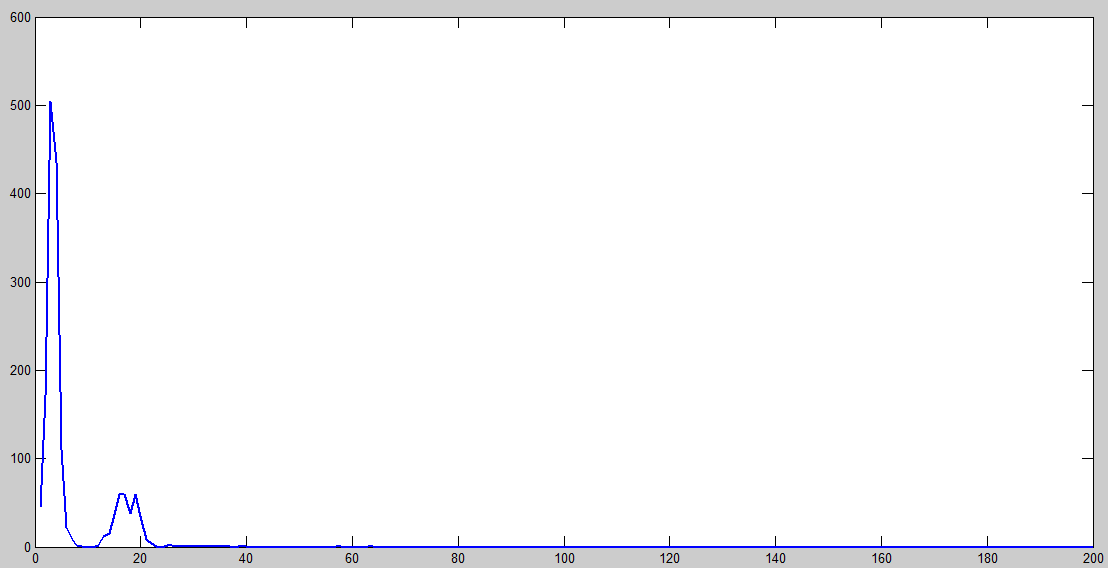
After experimenting on various documents we set the value of α to 10.

****

1. **Word Blob Formation:**

After segmentation of text lines, we need to identify single characters from each line for operator detection. Gap between two characters from each line is calculated and it is projected on a two dimensional plane where X axis denotes gap amount and Y axis denotes the number of occurrences of each gap. Two properties are observed from this plot: gap between two characters is lesser than the gap between the last and first character of two consecutive words and number of characters is more than the number of words in any document. Hence, the gap where the first local peak is found denotes the character gap which has the maximum occurrence and the gap where the second local peak occurs denotes the word gap with maximum occurrence. So, the local minimum between these two peaks gives us the maximum character gap.

We close the document image with this gap distance, and the word blob is formed. This is perfect for English documents. For Bangla documents, due to the headline of every word, all the characters are connected and no gaps exist between them. Hence, the X value corresponding to first peak denotes the word gap with maximum occurrence and the length of the structural element for closing operation is taken some value lesser than this X value.



1. **Operator Recognition**

After word blob formation, operators have to be detected. In this proposed method, we have detected operators {+, -, X, =, ∏, ∑, →, ―} to identify the lines with equations. It can be concluded that any line containing embedded or displayed equations must have at least one of these operators.

**Detection of operators:**

After the blobs are formed, they are labeled. The region corresponding to each blob is considered from original image and number of connected component present in that region is counted. Count more than one indicates it’s a word and not an operator. This eliminates the words. Furthermore, we have also taken into the account the Euler number which will be 1 for the operators we want to detect and size of which will not be less than average character height and width. This helps elimination of some single characters like ‘a’, ‘A’ and full stops. We then took horizontal and vertical projection profile, height and width, and for all the remaining symbols.

The features are matched against certain constraints to determine the correct symbols. Table below shows the constrains corresponding to the operators we want to detect

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operator** | **Horizontal Projection Profile** | **Vertical Projection Profile** | **Height: Width** | **Histogram of first white pixel when scanning from left-to-right top-to-bottom** |
| + | Peak near (row/2) | Peak near (column/2) | Near 1 | Smooth except around (row/2) |
| - | Constant | Inverted ‘V’ | Less than 3/5 | Constant |
| X | Almost constant | Almost constant | Near 1 |  |
| − | Constant | Inverted ‘V’ | Less than 3/5 but width >median width and height<median height | Constant |
| ∑ | Highest peak near start and end of row | Value till (column/3) > rest | Greater than 1  Height > median height and  Width > median width | Increasing till (row/2), then decreasing |
| Π | Peak near the start | Peak near start and end of row | Greater than 1  Height > median height and  Width > median width | Constant except for first few rows |

**Detecting ‘=’:** Once a ‘-‘is detected, we scan up and down the detected ‘-‘till the width of detected ‘-‘. If any white pixel is detected in that range, we select the entire white pixel region and check it for ‘-‘detection. If the region of white pixel comes out as ‘-‘, we fill the entire scanned region with white pixel to prevent detecting the second ‘-‘of equal as operator again.

1. **Equation segmentation**

First, the lines which contain at least one of our detected operators are segmented out. Now division operator has numerator and denominator which are often detected as separate lines. To correct this, we scan the top and bottom part division operator

1. **Equation segregation – inline or displayed**
2. **Displayed Equation Classification**