

# **Automated Segmentation, Classification And Generation of Searchable PDF Format of Chemical Equations from Document Images**

Prerana Jana  
Exam Roll : 111105023

Anubhab Majumdar  
Exam Roll : 111105049

Department of Computer Science And Technology  
IEST Shibpur

May 6, 2015

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Previous Works . . . . .	3
1.2	Motivation . . . . .	4
<b>2</b>	<b>Proposed Work</b>	<b>6</b>
2.1	Pre-Processing . . . . .	6
2.2	Equation Segmentation . . . . .	6
2.2.1	Text line Segmentation . . . . .	7
2.2.2	Creation of word blobs in Text lines . . . . .	8
2.2.3	Operator identification . . . . .	9
2.2.4	Segmentation of DE zones . . . . .	11
2.2.5	Experimental Results . . . . .	12
2.2.6	Error Analysis . . . . .	13
2.3	Classification of segmented DE zones . . . . .	13
2.3.1	Experimental Results . . . . .	15
2.3.2	Error Analysis . . . . .	16
2.4	Auto correction of chemical compounds and the equation . . .	16
2.4.1	Coefficient Extraction . . . . .	18
2.4.2	State separation . . . . .	19
2.4.3	Auto correction of the formula unit . . . . .	19
2.4.4	Auto correction of the entire equation using Context Table . . . . .	20
2.4.5	Experimental Result . . . . .	22
2.4.6	Error Analysis . . . . .	24
<b>3</b>	<b>Conclusion</b>	<b>25</b>

# Chapter 1

## Introduction

Segmentation of mathematical equations from document images is already a major research area for improved performance of OCR systems. Though chemical equations are also sharing similar spatial properties as that of non-chemical equations, efforts to segment them are yet to be explored. Also, PDF format of scanned document images is not searchable. OCR tries to remedy this adversity by converting images or PDF files into editable and searchable data, but it has its own limitations in presence of equations - both mathematical and chemical. This project paper presents a method for segmenting and identifying chemical and any other equations in heterogeneous document images that may contain graphics, tables, text and classifying them into two categories - chemical and other equations. This study, a first of its kind, as far our knowledge goes, not only improves the OCR performance but leads to creation of chemical database and formation of bond electron matrix from chemical equations or formulae. In our proposed method we extracted the equations using morphological operators and histogram analysis; and to evaluate the performance of proposed method, the extracted equations are classified using an open source OCR engine. The effectiveness of the proposed method is demonstrated by testing it on 152 document images. Test results show an accuracy of 97.47% and 97.425% for segmentation and classification, respectively. Also, we present a novel method for automated generation of searchable PDF format of segmented chemical equations from scanned document images by performing chemical symbol recognition and auto-correction of chemical reactants. We use existing OCR systems, pattern recognition, contextual data analysis and a standard L<sup>A</sup>T<sub>E</sub>X package to generate the chemical equation in searchable PDF format. The effectiveness of the proposed method is demonstrated by testing it on 240 document images.

## 1.1 Previous Works

Large number of documents are being digitized today for the purpose of archival analysis, transmission and browsing. The existing OCR systems show high accuracy in interpreting text portions but fail to properly process other components like graphics, halftones, chemical and mathematical equations. Also, the existing search engines take the text-based keywords for retrieving documents overlooking chemical/mathematical equations from scientific documents.

A few studies [20, 2, 22] are directed toward math-symbol or math equation recognition assuming that the math-zones are already marked. Though, symbol recognition is a part of OCR activity; when it is applied to the non-segmented mixed material (text with math-zone and others) computation is expensive and success far from satisfactory. We on the other hand contend that a better approach is to segment the mathematical/chemical equations from the mixed material thereby helping the future OCR activity to focus its processing only on specific content. In this paper we propose fully automated segmentation technique for extracting mathematical/chemical equations exploiting spatial distribution of black pixels on a white background and subsequently classifying them using an open source OCR.

A number of work has been done over the past decade to detect and extract the mathematical equations present in heterogeneous document images. Fateman et al. [4] proposed a scheme which utilised character size, font information etc. to identify all connected components. Two bags, namely *text* and *math* are defined. The *text* bag is used to keep all letters and italic numbers; whereas the *math* bag collects punctuation, special symbols, Roman digits, italic letters, lines and dots. The *math* bag objects are then grouped together according to their spatial proximity. Grouping of items in text bag is done next followed by review and correction to move isolated items to their proper destinations. Math segmentation is done in [5] through physical and logical segmentation using spatial characteristics of the math zone as well as identifying some math-symbols. The document is segmented to characters, words, lines and blocks by physical segmentation. The logical segmentation process that follows consists of two steps; first the displayed math is detected by identifying their usual centered position and in the next step in-line maths is detected by identifying special symbols.

Kacem et. al. [6] extracted the equations using fuzzy logic by detecting mathematical operators. Their method was tested on a dataset consisting of 300 expressions and the success rate is about 93%. As some of the operators like '+', '-', and '()' do appear in chemical equations as well, it leads to the mis-classification of chemical equations as mathematical equations reducing

the success rate. A similar method has been proposed in [7] to segment the mathematical expression in printed documents. The statistical approach taken by Garain [9] on the corpus of 400 pages to differentiate normal text lines and lines containing equations/expressions is on the basis of their white spacings which are usually larger in math-equation than the normal text. However, the chemical equations in the documents bear the same property. Jin et. al.[11] proposed a similar method to extract displayed formulas using Parzen classifier. Drake and Baird [12] came up with a graphical approach; similarly Guo et. al.[13] developed a Gaussian mixture model to describe spatial relationships between sub-components of a math expression. Another method to check text style (regular, italic, bold) at the character level has been proposed in [10]. Garain [8] proposed a method to segment the displayed and embedded mathematical formulas from the documents using a bunch of features. The method is tested on a dataset of 200 images containing 1163 embedded and 1039 displayed expressions and the success rate is 88.3% and 97.2% respectively for embedded and displayed expressions. A method proposed by Chu and Liu [14] used features based on centroid fluctuation information on non-homogeneous regions to detect displayed and embedded formulas.

There are some methods that are used to reconstruct chemical molecules from scanned image. They have used chemical datasets. Algorri et al. [24, 25] proposed a system that reconstruct chemical molecules from scanned document. They have used connected component analysis and their own vectorisation algorithm for character recognition. Connected components that are not recognised by the OCR engine, are used to produce a graph of vectors. A rule based approach reconstructs the molecule description from the vector graph and the character information. ChemReader [26] starts with connected component. Alphanumerics are recognised using the GOCR open source OCR tool. Graphical components are identified using Hough transforms, corner detection and other bespoke algorithms.

## 1.2 Motivation

In a nutshell, in all the above methods emphasis is given only in mathematical equation; and in the eventual segmentation/classification, chemical equations would automatically be included as a part of mathematical (or other) equations thereby reducing the success rate of the segmentation and effectiveness of the subsequent classification, if any.

Considering the possible applications of the segmentation of chemical equations, we see that the recent development in the field of chemo-informatics

requires precise identification of chemical equations amongst a myriad collection of chemical and non-chemical formulas/equations. This can be important for various tasks like creation of chemical database as well as to obtain bond-electron matrix from a given chemical equation, etc. The proposed work embodied in this paper is motivated by the aforementioned needs.

# Chapter 2

## Proposed Work

### 2.1 Pre-Processing

The work starts with heterogeneous binary images that may contain text and graphics. The graphics, tables and headings are extracted out from the heterogeneous documents keeping the math-zone/chemical-zone, if any, along with the segmented and skew corrected text following popular and robust methods described in [17] and [15]. We did not consider in-line expressions because chemical equations are rarely present in that form. Since, our main motivation was to classify chemical and non-chemical equations, we restrict our study to displayed equations only.

### 2.2 Equation Segmentation

Most of the elements in the displayed equations (DE) have little difference from the normal text. Naturally, the segmentation depends on a couple of rules formed by observing the general spatial appearance of displayed-equations in common technical documents including journals. This is carried out by sampling 152 scanned pages containing mathematical/chemical equations in different possible forms. The following are the general spatial characteristics of the DE zones:

- Subscripts and superscripts are frequently present in DE zones.
- Math expressions are often written using 2-3 row height; this happens to accommodate subscripts and superscripts leading to vertical overlap of characters that is absent in normal text lines.

- In math expressions the characters and symbols are less dense in comparison to normal text lines.
- Presence of different operators in DE zones.
- Presence of a horizontal line separating the numerator and denominator portions is common.
- The DE zones are generally aligned, of which central alignment is most frequent.

Segmentation of the DE zone starts with the removal of very small components, like dots of i, j and small punctuation marks like comma, period etc. by using component labelling and a suitable area threshold.

Major steps that follow for DE segmentation are given below.

1. Text line segmentation
2. Blob formation using morphological tools
3. Operator identification
4. DE zone segmentation

The details of the above steps are as follows:

### 2.2.1 Text line Segmentation

To detect DE zones, text lines have to be segmented first from which the operators are identified to determine whether a text line is a displayed equation or not. We have taken the horizontal projection profile of the document page to segment the text line. A document page and its horizontal projection profile are shown in Fig. 2.1

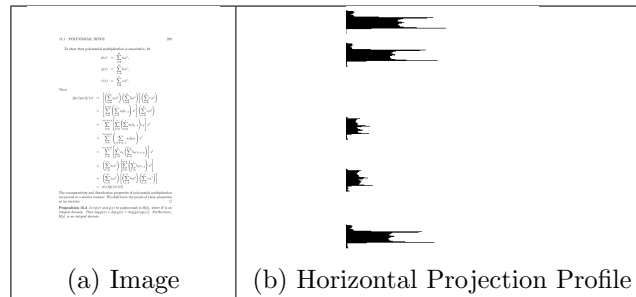


Figure 2.1: A document page and its horizontal projection profile



### 2.2.2 Creation of word blobs in Text lines

This is done by merging the characters in a word. Such character coalescing process depends on the accuracy in detecting the normal character gap and the gap between the consecutive connected components in that text line.

The mathematical formulation for blob formation is as follows. Consider a binary image  $I_{P \times Q}$ , which consists of connected components  $C_k (k = 1, 2, \dots, P)$ , as defined in standard text [16] with their usual meanings. Let  $L(C_k)$ ,  $R(C_k)$ ,  $T(C_k)$  and  $B(C_k)$  be the four corner points of the  $k_{th}$  connected component in left, right, top and bottom directions respectively.

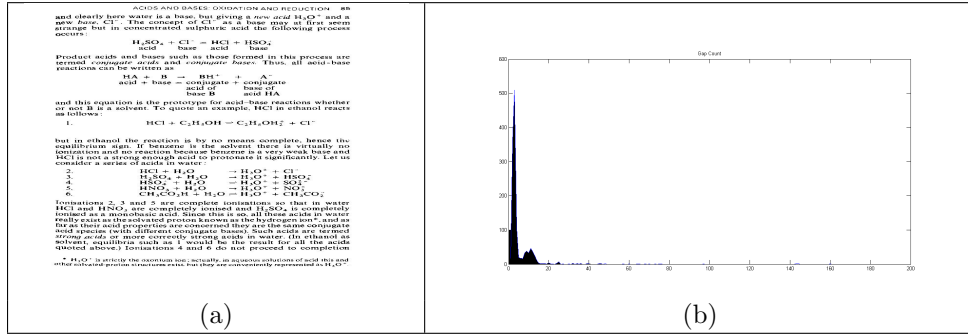


Figure 2.2: Example of a page and histogram. (a) Document page (b) Histogram of the preprocessed page.

Let  $F$  be a function which ensures that the two connected components lie in the same text line. Then  $F$  may be represented as

$$F(C_a, C_b) = \begin{cases} 1 & \text{if } (T(C_a) \leq B(C_b) \text{ AND } B(C_a) \geq T(C_b)) \\ 0 & \text{otherwise} \end{cases}$$

Cluster formation requires information on inter-word gap.

The distance function,  $D$ , obtained from the histogram  $H_1$ , is defined for computing the horizontal distance between any two consecutive connected components  $C_a$  and  $C_b$ , as shown below

$$D(C_a, C_b) = L(C_b) - R(C_a)$$

where  $b = \min_x \{L(C_x) - R(C_a)\}$  such that  $(F(C_a, C_x) = 1 \text{ AND } L(C_x) > R(C_a))$ . The histogram  $H_1$  registers the gap between two consecutive characters. It may be noted that there may be more than two distinct humps in  $H_1$ , first one represents the character gaps in words and the second one represents the normal word gaps in text lines. An example page and corresponding histogram is shown in Fig. 2.2(a) and (b).

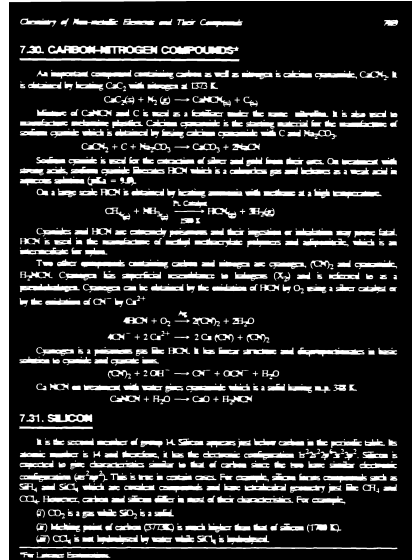


Figure 2.3: Example of blob formation. Clusters formed from a portion of the image shown in Fig. 2.2(a).

Our intention is to find out character gaps in running texts of a document page so that we could combine the consecutive characters into a single blob. Hence, we consider the upper boundary ( $v$ ) of the first hump as the length of structuring element. Morphological close operation with a structuring element of area ( $v \times 1$ ) will form the blobs denoted as  $V_w$  (where  $w = 1, 2, \dots, Q$ ). The blob formation will be dictated by the following two conditions:

1. If there are two connected components  $C_m$  and  $C_n$  ( $1 \leq m, n \leq P$ ) obeying the relations

- $F(C_m, C_n) = 1$
- $D(C_m, C_n) \leq v$

then  $C_m$  and  $C_n$  should belong to the same blob.

2.  $V_a \cap V_b = \emptyset \quad \forall a, b \mid (1 \leq (a, b) \leq Q \text{ AND } a \neq b)$

### 2.2.3 Operator identification

We have considered the set of operators that is commonly used both in chemical equations as well as mathematical equations to fulfil our aim to classify displayed zones containing chemical and non-chemical equations.

Operators	horizontal projection	vertical projection
+		
—		
→		

Figure 2.4: Operators and their horizontal and vertical projection profiles

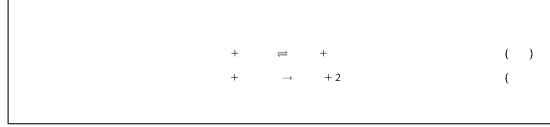


Figure 2.5: Sigle components extracted from the word blobs shown in Fig. 2.3

The remaining components in the blob image are operators along with some alphanumerics like ‘ $a'$ ’, ‘ $A'$ ’, ‘ $'$ ’, etc. The logical AND operation is performed between the blob image and the original image. The Euler number of the operators that we have considered is 1 (one) and based on this feature some of the alphanumerics are discarded. This image is denoted by  $I_s$ . Width ( $w$ ) and height ( $h$ ) of each component in  $I_s$  are determined. The components are divided into two classes based on the ratio of  $w, h$ ; (i) *thin class* ( $\frac{w}{h} > 2$ ) and (ii) *regular class* ( $0.8 < \frac{w}{h} < 1.3$ ). Some instances of the *thin* and *regular classes* are ( $-$ ,  $\rightarrow$ ,  $\leftarrow$ ,  $\leftrightarrow$ ,  $\rightharpoonup$ ,  $\leftrightsquigarrow$ ,  $\sim$ , etc.) and ( $+$ ,  $\bullet$ ,  $\times$ , etc) respectively.

The number of *on pixels* ( $N_p$ ) and number of *off pixels* ( $N_w$ ) within each bounding box of the thin components are calculated. Morphological opening operation is performed on *thin class* with a line like structuring element (SE) of length  $\frac{w}{2}$ . If the output of the opening operation has a single component, then thinning is performed on the corresponding original component. Next, the number of end points of each thinned component is checked, if the number of end points is two and  $\frac{N_p}{N_w} > 0.8$ , then the component under test is a ‘ $-$ ’ sign. If the number of end points is greater than two, then element is an arrow. We need to identify the direction of the arrowhead for its correct representation in  $\text{\LaTeX}$  file. The direction of arrowhead is identified by measuring the height of the arrow element near its two ends.

From the *regular class* our objective is to identify the ‘ $+$ ’ sign and it is detected using the following rules:

- (i) Horizontal and Vertical Projection Profile of each component is considered;
- (ii) Ratio of *maxProj*,  $D$  and ratio of *maxIndex*, ( $M_l$ ) are obtained where *maxProj* is the maximum horizontal or vertical projection value, *maxIndex* is the index of *maxProj* closest to  $M_l$  and  $D$  is width (height) for horizontal

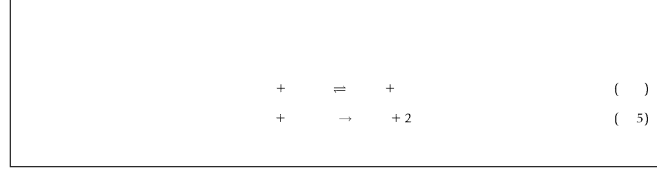


Figure 2.6: After removal of alphanumerals based on Euler number from the image shown in Fig. 2.5

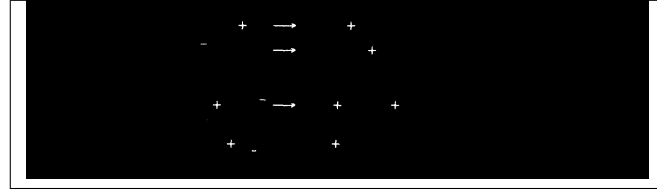


Figure 2.7: Operators extraceted from the image shown in Fig. 2.6

(vertical) profile. Here,  $M_l$  is  $y(x)$  coordinate of the middle point for horizontal (vertical) profile. If  $0.9 < \frac{\max Proj}{D} \leq 1$ ,  $0.9 < \frac{\max Index}{M_l} \leq 1.2$  and  $\frac{N_w}{N_p} > 3$ , then the component under test is a '+' sign. Where  $N_w$  and  $N_p$  are number of *on pixels* and *off pixels* along the diagonal direction of the bounding box of the component. All the threshold values used here are selected based on our empirical study on 234 images.

To detect '=' or '⇒' one extra step is required. The operators having  $f_a \leq 0.6$  are considered to be thin symbols ( $-$ ,  $\leftarrow$ ,  $\rightarrow$ ,  $\leftrightarrow$ ). For each symbol denoting thin operator, a rectangular mask is placed below the symbol to check if there is another one within the mask; if present the two thin operators are considered to form an '=' or '⇒' sign. Let the length of the thin operator be  $l$ . The area of the mask is  $(l \times l/2)$ .

The horizontal line separating the numerator and the denominator is identified by its length which is greater than the median length of the operators. Two windows are placed above and below the separating line to merge all the components within the windows with the separating line to form a single logical line. Otherwise, they would be treated as three consecutive text lines and we will not be able to associate the intermediate math-symbols ('+', '-', '=') to a single expression. The area of the window is (length of the separating line)  $\times$  (twice the median height of the text lines).

#### 2.2.4 Segmentation of DE zones

Initially, all the text lines consisting of at least one operator are considered as candidate displayed equations (CDE). The operators are eliminated from

CDE. The upper boundary ( $u_v$ ) of the second hump of the histogram  $H_1$  is obtained which represents the word gaps in the text line. For each CDE zone Run Length Smoothing Algorithm in horizontal direction ( H-RLSA) is carried out. If the distance between two neighbouring components is less than  $u_v$ , it means they belong to a same word and thus are merged by H-RLSA. H-RLSA has a similar effect as of dilation of black areas in horizontal direction. The characters in a word are dilated and get linked/connected to the other characters of the same word. The output of H-RLSA is shown in Fig. 2.8.





NaH + H <sub>2</sub> O → H <sub>2</sub> + NaOH				(10.1)
(a)				
NaH	H <sub>2</sub> O	H <sub>2</sub>	NaOH	(10.1)
(b)				
				(10.1)
(c)				

Figure 2.8: The output of H-RLSA on portion of an image (a) a part of an image; (b) same part without operators; (c) result of H-RLSA operation

The equation number is common in the displayed equation zone. This has to be removed because for each CDE we have counted the number of operators and corresponding other components in the output of H-RLSA. If the number of components  $\leq 2 \times$  number of operators, then the CDE is considered as a displayed equation; otherwise some embedded formulas/equations may exist in the line. To eliminate the equation number from the output of H-RLSA the operators are moved here and the component analysis is done. From both ends distance ( $d$ ) (see Fig.2.9) between the first two consecutive components is measured and if  $d > 5 \times u_v$ , then the first component from the end is considered as equation number and is removed.

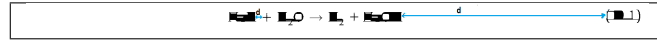


Figure 2.9: Example of a equation number present in DE zone

### 2.2.5 Experimental Results

We have implemented our algorithm in MATLAB 8.3.0.532 (R2014a) in a PC (Intel(R) Core(TM) i5-3337U CPU @ 1.80GHz running Windows 8). The proposed method has been tested on a dataset consisting 234 document pages. Out of 234 pages 50 pages are taken from ICDAR 2013 Math-zone

segmentation datasets and other document pages are scanned from different Mathematics and Chemistry books.

Table 2.1: Summary of Experimental Results

# Total Images	234
# Total DEs	1390
DE segmentation accuracy	98.63%

## 2.2.6 Error Analysis

### Segmentation error

Some chemical equations have some reactants/symbols (such as  $\Delta$ ) just on top or bottom of the arrow and in some cases they enter the bounding box of the arrow. When we extract only single characters from the word blob information as mentioned in 2.2.3, we do not get the arrow as it is no longer treated as a single character within its bounding box. Here the operator, arrow does not get detected and the ratio between number of operands and number of operators exceeds 2. So, this is segmented as an embedded equation. See. Fig. 2.10(a).

$\text{Ca(HCO}_3)_2 \xrightarrow{\Delta} \text{CaCO}_3 \downarrow + \text{CO}_2 + \text{H}_2\text{O}$ <p>(a)</p>
<p><b>Proof of Proposition 3.</b> Maximization of bank 1's <i>ex ante</i> expected net profits in Equation (4), <math>E[\Pi_1] = (2 - \lambda) \frac{1}{2} (2\phi - 1) - (\phi - \frac{1}{2})^2 + \lambda\phi(1 - \phi)(\bar{p}R - 1)</math>, with respect to <math>\phi</math> yields <math>\phi^* = \frac{3 + \lambda(\bar{p}R - 2)}{2 + 2\lambda(\bar{p}R - 1)}</math> from the FOC</p> <div style="border: 1px solid green; padding: 5px; margin: 10px auto; width: fit-content;"> <math display="block">\frac{\partial E[\Pi_1]}{\partial \phi} = 3 - \lambda + \lambda(1 - 2\phi)(\bar{p}R - 1) - 2\phi = 0</math> </div> <p>Since <math>\lambda \in [0, 1]</math>, we can always choose <math>\bar{p}R &lt; M</math>, for sufficiently small <math>M</math>, such that the optimal</p> <p style="text-align: center;">(b)</p>

Figure 2.10: Examples of segmentation error

In a few cases, text lines consisting of embedded expressions are identified as displayed equations (see Fig. 2.10(b)), but in those cases the text lines contain more mathematics than normal text. However, identification of mathematics intensive text lines as mathematical displayed equation is not a severe error.

## 2.3 Classification of segmented DE zones

Each segmented displayed equation is divided into three zones; namely upper zone, middle zone and lower zone (see Fig 2.11). To identify the three zones

of a DE zone, uppermost and lowermost co-ordinates of each connected component below the same DE zone are also obtained. The median of uppermost coordinate, and median of lowermost co-ordinate of such components in DE zone are computed. A horizontal line, called the baseline, is drawn through the median of lowermost coordinates of components and this baseline separates the middle zone and lower zone of DE zone. Similarly, the median of uppermost co-ordinate of the components in the DE zone generates a horizontal line. This horizontal line, called top line, separates the middle and upper zones of the DE zone.

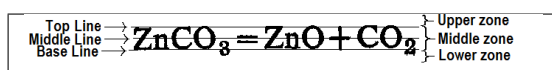


Figure 2.11: Example of 3 zones present in a DE zone

The subscripts in a DE zone belong to lower-half of the middle zone and lower zone whereas the superscripts belong to upper zone and upper-half of the middle zone. Based on the location of the components in a DE zone we have detected the subscripts and superscripts and removed from the DE zone. The operators are also removed from DE zone.

Now, each displayed equation is an input to an OCR of MATLAB R2014a. The OCR returns each DE zone as a text string. We made a dictionary out of all the elements in the periodic table. An important observation is that an element always starts with a capital letter. Using this property, an element can be expressed by a regular expression  $[A-Z][a-z]^*$ . It means an element's symbol starts with an upper case letter and may or may not have one or more lower case letters. We have designed a parser to extract the sub-strings matching the above-mentioned regular expression with the following grammar:

$$\begin{aligned} start &\rightarrow capital.follow \\ follow &\rightarrow small.follow \mid \epsilon \\ capital &\rightarrow A|B|\dots|X|Y|Z \\ small &\rightarrow a|b|\dots|x|y|z \end{aligned}$$

The working principle of the parser is depicted in Fig. 2.12. Each of the substrings returned by the parser is matched against the aforesaid dictionary and if it is a positive match then that substring is considered as a symbol of the chemical element. Let us consider, the number of substrings extracted from the OCR output by the parser is  $n$  and the number of positive matches with the aforementioned dictionary is  $m$ . If  $m:n$  ratio is more than a threshold

value  $\beta$  then this DE is considered as a Chemical Equation. This threshold ( $\beta$ ) is set to 0.7 by running our algorithm on our dataset containing 733 segmented displayed equations. The reasons for  $\beta$  not being 1 are i) limitation of the OCR we used; ii) presence of broken and touching characters in the DE zone.

The  $\uparrow$  and  $\downarrow$  are frequently used in chemical equation to represent the state of compounds and are thus important to detect. Let each chemical equation be denoted by  $C_e$ . Each  $C_e$  zone is AND with  $I_s$  which produces an output  $I_o$ .  $I_o$  may contain operators, single character elements and ( $\uparrow$ ,  $\downarrow$ ). The operators are removed from  $I_o$  as they are already identified. For the rest of the components in  $I_o$ , we have checked whether the component is the starting character of the chemical equation, if not, then its immediate left neighbour in  $C_e$  is checked. If the left neighbour is not an operator, then then component under test is  $\uparrow$  or  $\downarrow$ . The arrowhead direction is determined in the same way as right/left arrow.

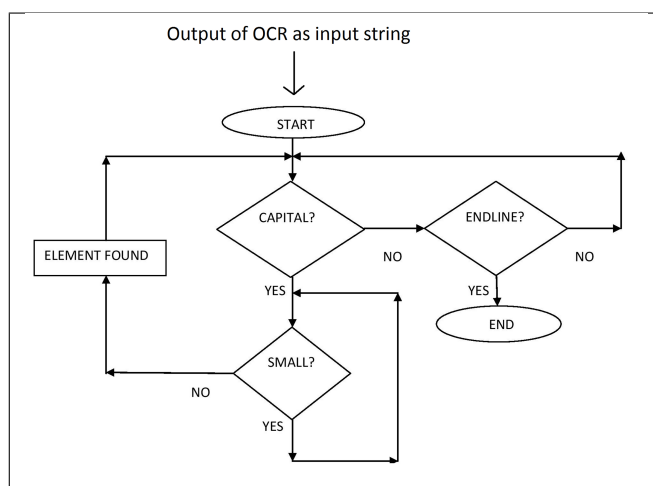


Figure 2.12: Working flow chart of the parser

### 2.3.1 Experimental Results

We have implemented our algorithm in MATLAB 8.3.0.532 (R2014a) in a PC (Intel(R) Core(TM) i5-3337U CPU @ 1.80GHz running Windows 8). The proposed method has been tested on a dataset consisting 234 document pages. Out of 234 pages 50 pages are taken from ICDAR 2013 Math-zone segmentation datasets and other document pages are scanned from different Mathematics and Chemistry books.



Classification outputs for some of the cases are shown in Fig. 2.13. In the figure the red and the green bounding boxes indicate chemical and non-chemical expressions respectively.

Table 2.2: Summary of experimental results

Actual \ Classified As	Chemical	Other
	Chemical	Other
Chemical	98.83%	1.17%
Other	1.17%	98.83%

### 2.3.2 Error Analysis

#### Classification error

In some cases if the operands of a chemical equation have Alkyl or Halide group, they are denoted as R and X respectively. But these symbols are not present in the periodic table. Hence, when the substrings from the OCR output are searched in the dictionary, it comes back negative. The equation is detected as a non-chemical one. See 4th equation bounded with a green rectangle in Fig. 2.14.

## 2.4 Auto correction of chemical compounds and the equation

Next, auto correction is performed based on the chemical context as existing OCRs can not produce perfect result in case of chemical equations. The output of H-RLSA (see Fig. 2.8 (d)) is taken as input here. Each character within a word blob is an input to the the OCR and the corresponding output is stored in a cell and these cells form a string,  $S_{chemical}$  for each word blob. For each superscript and subscript, '^' and '\_' are inserted before them respectively in  $S_{chemical}$ .

First, an error map is created based on the observation of OCR outputs of 280 chemical equations consisting of 1022 compounds (see Table 2.3). Next, this table is stored into a hash map  $H$  where the key is the OCR output and its value is the possible input set. For example, if '8' is an erroneous OCR output for inputs 'g', '3' and 'a' (see Table 2.3) then, in the hash map  $H$ , key is '8' and its corresponding value is  $[g, 3, a]$ .

Each chemical compound in any equation (see Fig. 2.15) has the following format-  $[Coefficient]^{[0,1]}[Formula\ Unit][state]^{[0,1]}$ . Auto correction of the

<p>নিম্নে আংশিক সমীকরণ সাহায্যে সমীকরণের সামগ্রিক বিধানের কয়েকটি উদাহরণ দেওয়া হইল।</p> <p>(ক) কপারের হিফ্রা ও বস, উক্ত সালফিউরিক অ্যাসিডের বিক্রিয়ার সালফার ডাই-অক্সাইড উৎপন্ন হয়। অপর বিক্রিয়ায় পার্শ্ব কপার সালফেট ও জল।</p> $\begin{aligned} \text{H}_2\text{SO}_4 &= \text{SO}_2 + \text{H}_2\text{O} + \text{O} \\ \text{Cu} + \text{O} &= \text{CuO} \\ \text{CuO} + \text{H}_2\text{SO}_4 &= \text{CuSO}_4 + \text{H}_2\text{O} \\ \text{Cu} + 2\text{H}_2\text{SO}_4 &= \text{SO}_2 + \text{CuSO}_4 + 2\text{H}_2\text{O} \end{aligned}$ <p>(খ) ম্যাগনেসিয়াম ডাই-অক্সাইড, সোডিয়াম ক্রোমাইট ও গ্যাস সালফিউরিক অ্যাসিড মিশ্রণ উত্তপ্ত করিলে ম্যাগনেসিয়াম সালফেট, ক্রোমিন, সোডিয়াম বাই-সালফেট ও জল উৎপন্ন হয়।</p> $\begin{aligned} 2\text{NaCl} + 2\text{H}_2\text{SO}_4 &= 2\text{NaHSO}_4 + 2\text{HCl} \\ \text{MnO}_2 + \text{H}_2\text{SO}_4 &= \text{MnSO}_4 + \text{H}_2\text{O} + \text{O} \\ 2\text{HCl} + \text{O} &= \text{H}_2\text{O} + \text{Cl}_2 \\ 2\text{NaCl} + \text{MnO}_2 + 3\text{H}_2\text{SO}_4 &= 2\text{NaHSO}_4 + \text{MnSO}_4 + 2\text{H}_2\text{O} + \text{Cl}_2 \end{aligned}$ <p>(ই) অ্যানিড-সূত্র হাইড্রোজেন পার-অক্সাইডের সহিত পটাসিয়াম পারম্যাংগেট বিক্রিয়া করিয়া গ্যাসীয় অক্সিজেন উৎপন্ন করে। অপরশর বিক্রিয়ায় পার্শ্বগুলি হইল ম্যাগনেসিয়াম সালফেট, পটাসিয়াম সালফেট এবং জল।</p> $\begin{aligned} 2\text{KMnO}_4 &= \text{K}_2\text{O} + 2\text{MnO} + 5\text{O} \\ \text{K}_2\text{O} + \text{H}_2\text{SO}_4 &= \text{K}_2\text{SO}_4 + \text{H}_2\text{O} \\ 2\text{MnO} + 2\text{H}_2\text{SO}_4 &= 2\text{MnSO}_4 + 2\text{H}_2\text{O} \\ 5\text{H}_2\text{O}_2 &= 5\text{H}_2\text{O} + 5\text{O} \\ 5\text{O} + 5\text{O} &= 5\text{O}_2 \\ 2\text{KMnO}_4 + 3\text{H}_2\text{SO}_4 + 5\text{H}_2\text{O}_2 &= \text{K}_2\text{SO}_4 + 2\text{MnSO}_4 + 8\text{H}_2\text{O} + 5\text{O}_2 \end{aligned}$ <p>এইরূপ পদ্ধতি প্রয়োগে জারণ বিজারণ, অ্যানিড-কার প্রথমে প্রকৃতি বিধে জানা যাওয়া বিশেষ দরকার।</p> <p><b>রাসায়নিক সমীকরণের তাৎপর্য (Significance of a chemical reaction) :</b></p> <p>রাসায়নিক সমীকরণ মাত্রেরই গুণগত (Qualitative) এবং পরিমাণগত (Quantitative), এই দুই রকম তথ্য প্রকাশ করে। সমীকরণ হইতে রাসায়নিক বিক্রিয়ায় পরস্পরের কি পরিবর্তন হইল, পরিবর্তনের ধর্মসমূহ কি কি পার্থক্য গঠিত হইল ইত্যাদি বিস্তারিত বোঝা যায়, যেমন হইল হারা কোন কোন পদার্থের কি পরিমাণ পরিবর্তিত হইল কি পরিমাণে নতুন পদার্থ উৎপন্ন হয় তাহাও জানা যায়। সমীকরণে পরস্পরের নিত্যতাও, ভাগটনের পরমাণুবাধ প্রকৃতির সূত্র কথাগুলি সর্বত্র রক্ষিত হয়। নিম্নে উদাহরণ দ্বারা সমীকরণের পূর্ণ তাৎপর্য বুঝানো হইল।</p>	<p>130 উচ্চ মাধ্যমিক রসায়ন</p> <p>অতঃপর, অ্যামরন (অস) এর তুল্যাক্ষ ভাৰ = <math>\frac{55.85}{2} = 27.925</math></p> <p>এবং অ্যামরন (ইক) এর = <math>\frac{55.85}{3} = 18.616</math></p> <p>পক্ষাঙ্করে বলা যায়, কোন মৌলের তুল্যাক্ষভার মৌলটি বিক্রিয়ায় যে ভাবে অংশ গ্রহণ করে তাহার উপর নির্ভর করে। নিম্নোক্ত সমীকরণ হইতে ইহা স্পষ্ট বুঝা যাইবে।</p> <p>(i) <math>\text{Fe} + 2\text{HCl} = \text{FeCl}_2 + \text{H}_2</math> (ii) <math>2\text{Fe} + 3\text{Cl}_2 = 2\text{FeCl}_3</math></p> <p>উপরের সমীকরণ দুইটির পরিমাণগত দিক বিবেচনা করিলে দেখা যায় প্রথম বিক্রিয়ার (i) 55.85 ভাগ অ্যামরন হাইড্রোক্লোরিক অ্যাসিডের সহিত বিক্রিয়ায় 2.016 ভাগ হাইড্রোজেন প্রতিস্থাপিত করে অথবা 70.92 ভাগ ক্লোরিনের সহিত যুক্ত হইয়া ফেরাস ক্লোরাইড গঠন করে। অতঃপর সজ্ঞাহুসারে অ্যামরনের (অস) তুল্যাক্ষ ভাৰ <math>\frac{55.85}{2} = 27.925</math>। একইভাবে দ্বিতীয় বিক্রিয়া (ii) হইতে যেখানে যায়, 18.616 ভাগ অ্যামরন 35.46 ভাগ ক্লোরিনের সহিত যুক্ত হইয়া ফেরিক ক্লোরাইড যৌগ সৃষ্টি করে। <math>\therefore</math> 18.616 সংখ্যাটিই ইক অ্যামরনের তুল্যাক্ষভাৰ।</p> <p><b>পারমাণবিক গুরুত্ব নির্ণয়ের রাসায়নিক পদ্ধতি :</b></p> <p>অ্যাতোমোজো প্রকল্প গ্রন্থে দ্বারা ক্যানিডারো পদ্ধতিতে মৌলের পারমাণবিক গুরুত্ব নির্ণয় প্রণালী ইতিপূর্বে আলোচিত হইয়াছে। এখানে আরও দুইটি পদ্ধতি সম্বন্ধে স্পষ্ট হইল।</p> <p><b>ডুলং ও পেটিট সূত্র প্রয়োগ করিয়া :</b> মৌলিক পদার্থের পারমাণবিক গুরুত্ব ও উহার আপেক্ষিক তাপের (Specific heat) গুণককে পারমাণবিক তাপ (atomic heat) বলে। নানা পরীক্ষার দ্বারা ডুলং ও পেটিট (Dulong and Petit) প্রমাণ করেন (সাধারণ তাপমাত্রায়) যে কোন কঠিন মৌলের পারমাণবিক তাপ সকল সময় একই হয় এবং উহার পরিমাণ প্রায় 6.4। অর্থাৎ কঠিন মৌলের পারমাণবিক গুরুত্ব এবং আপেক্ষিক তাপের গুণকল সর্বদা 6.4 (প্রায়) হয়। ইহাই ডুলং ও পেটিটের সূত্র।</p> <p>পারমাণবিক গুরুত্ব = <math>\frac{6.4}{\text{আপেক্ষিক তাপ}}</math></p> <p>অতঃপর, কোন কঠিন মৌলের আপেক্ষিক তাপ নির্ধারণ করিতে পারিলে উহার আয়নিক পারমাণবিক গুরুত্ব জানা হইতে পারে।</p> <p><b>পারমাণবিক গুরুত্ব নির্ণয়ে ডুলং পেটিট সূত্রের সীমাবদ্ধতা :</b></p> <p>প্রথমতঃ, ইহা কেবল কঠিন মৌলের ক্ষেত্রেই ব্যবহৃত হইতে পারে। তদুপরি কার্বন, বোরন, সিলিকন, বেরিলিয়াম কঠিন মৌল হইলেও ইহাদের ক্ষেত্রে সূত্রটি বাটে না। এই হেতু প্রয়োগে পারমাণবিক গুরুত্ব ব্যর্থ বা সঠিক ভাবে নির্ণীত হয় না।</p>
<p>ACIDS AND BASES: OXIDATION AND REDUCTION 85</p> <p>and clearly here water is a base, but giving a new acid <math>\text{H}_3\text{O}^+</math> and a new base, <math>\text{Cl}^-</math>. The concept of <math>\text{Cl}^-</math> as a base may at first seem strange but in concentrated sulphuric acid the following process occurs:</p> $\text{H}_2\text{SO}_4 + \text{Cl}^- = \text{HCl} + \text{HSO}_4^-$ <p>acid base acid base</p> <p>Product acids and bases such as those formed in this process are termed <i>conjugate acids</i> and <i>conjugate bases</i>. Thus, all acid-base reactions can be written as</p> $\text{HA} + \text{B} \rightleftharpoons \text{BH}^+ + \text{A}^-$ <p>acid + base = conjugate + conjugate acid of base of base B acid HA</p> <p>and this equation is the prototype for acid-base reactions whether or not B is a solvent. To quote an example, HCl in ethanol reacts as follows:</p> $\text{HCl} + \text{C}_2\text{H}_5\text{OH} \rightleftharpoons \text{C}_2\text{H}_5\text{OH}_2^+ + \text{Cl}^-$ <p>but in ethanol the reaction is by no means complete, hence the equilibrium sign. If benzene is the solvent there is virtually no ionization and no reaction because benzene is a very weak base and HCl is not a strong enough acid to protonate it significantly. Let us consider a series of acids in water:</p> $\begin{aligned} 1. & \text{HCl} + \text{H}_2\text{O} \rightleftharpoons \text{H}_3\text{O}^+ + \text{Cl}^- \\ 2. & \text{H}_2\text{SO}_4 + \text{H}_2\text{O} \rightleftharpoons \text{H}_3\text{O}^+ + \text{HSO}_4^- \\ 3. & \text{HSO}_4^- + \text{H}_2\text{O} \rightleftharpoons \text{H}_3\text{O}^+ + \text{SO}_4^{2-} \\ 4. & \text{HNO}_3 + \text{H}_2\text{O} \rightleftharpoons \text{H}_3\text{O}^+ + \text{NO}_3^- \\ 5. & \text{CH}_3\text{CO}_2\text{H} + \text{H}_2\text{O} \rightleftharpoons \text{H}_3\text{O}^+ + \text{CH}_3\text{CO}_2^- \end{aligned}$ <p>Ionisations 2, 3 and 5 are complete ionisations so that in water HCl and <math>\text{HNO}_3</math> are completely ionised and <math>\text{H}_2\text{SO}_4</math> is completely ionised as a monobasic acid. Since this is so, all these acids in water really exist as the solvated proton known as the hydrogen ion*, and as far as their acid properties are concerned they are the same conjugate acid species (with different conjugate bases). Such acids are termed <i>strong acids</i> or more correctly strong acids in water. (In ethanol as solvent, equilibria such as 1 would be the result for all the acids quoted above.) Ionisations 4 and 6 do not proceed to completion</p> <p>* <math>\text{H}_3\text{O}^+</math> is strictly the oxonium ion; actually, in aqueous solutions of acid this and other solvated proton structures exist, but they are conveniently represented as <math>\text{H}_3\text{O}^+</math>.</p>	<p>Lemma 6.1. The number of exact matches of a given type in a data set is equal to the number of relaxed matches minus the number of exact matches of all predecessor types.</p> $\bar{X}(\mathbf{x}_i^M, \mathbf{S}_i) = \bar{R}(\mathbf{x}_i^M, \mathbf{S}) - \sum_{j \in \mathcal{P}_i} \bar{X}(\mathbf{y}_j^M, \mathbf{S})$ <p>where <math>\mathbf{y}_j^M \in \mathcal{M}_i : \mathbf{y}_j^M &lt; \mathbf{x}_i^M</math>.</p> <p>An intuitive way to see this lemma is that an exact match, is a relaxed match of the same class, minus those exact matches of predecessor classes.</p> <p>It is worth noting a trivial corollary of this:</p> <p>Corollary 6.2. <math>\bar{R}(\mathcal{M}_i(T)) = \bar{X}(\mathcal{M}_i(T))</math> (2)</p> <p>which follows obviously from lemma 6.1 since there are no <math>\mathbf{x}_i^M &lt; \mathcal{M}_i(T)</math> and therefore the subtracted term in the lemma vanishes.</p> <p>Lemma 6.3.</p> $\bar{R}(\mathbf{x}_i^M, \mathbf{S}_i) = \prod_{j=1}^{m(\mathbf{x}_i^M)} \bar{X}(\mathcal{M}_{j,i}(T), \mathbf{Y}_i)$ <p>where <math>m(i) = \#Y_i</math> and <math>Y_i</math> is an <math>m</math>-tuple constructed from the original <math>n</math>-tuple of sites <math>\mathbf{S}</math> using the matching class <math>\mathbf{x}_i^M</math>. <math>Y_i</math> is constructed such that <math>Y_i = (S_{i_1}, S_{i_2}, \dots, S_{i_m})</math> where <math>\{j_1, \dots, j_m\}</math> is the set of all indices of the <math>n</math>-tuple <math>\mathbf{x}_i^M</math> such that <math>x_{j_k} = i</math>.</p> <p>An example may help understand how <math>Y_i</math> is constructed. If <math>\mathbf{x}_i^M = (1, 1, 2, 3, 2, 1)</math> then <math>Y_i = (S_1, S_2, S_4, S_5, S_6, S_6)</math> and <math>Y_i = (S_1)</math>.</p> <p>Again, the proof of this lemma is not stated here. It can be thought of as breaking down a relaxed match into the component exact true matches on subsets of <math>\mathbf{S}</math>, which are necessary conditions for a set of observations to be a relaxed match of the given type.</p> <p>It is worth noting a trivial corollary of this.</p> <p>Corollary 6.4.</p> $\bar{R}(\mathcal{M}_i(T), \mathbf{S}) = \prod_{j=1}^n \#L(\mathbf{S}_j)$ <p>Proof. This should be obvious since for <math>\mathcal{M}_i(T) = (1, 2, \dots, n)</math> each set of sites <math>Y_i</math> consists of exactly one site <math>i</math>. Since <math>\mathcal{M}_i = \{i\}</math> then there is only one matching class for each of the sites and <math>\bar{X}(\mathcal{M}_i(T), \mathbf{Y}_i) = \#L(\mathbf{S}_i)</math>. <math>\square</math></p> <p>Lemma 6.5.</p> $\bar{X}(\mathcal{M}_i(T), \mathbf{S}) = \bar{X}(\mathcal{M}_i(T), \mathbf{C}(\mathbf{S})) - \sum_{j \in \mathcal{P}_i} \bar{X}(\mathbf{x}_j^M, \mathbf{S}) p(\bar{H}(\mathbf{x}_j^M))$ <p>where <math>\mathbf{x}_j^M \in \mathcal{M}_i : \mathbf{x}_j^M &lt; \mathcal{M}_i(T)</math></p>

Figure 2.13: Samples of classification result

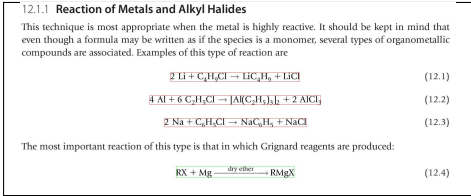


Figure 2.14: An Example of classification error.

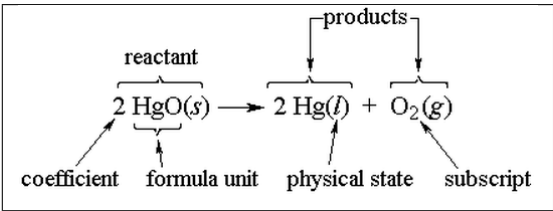


Figure 2.15: Different components of a chemical equation.

OCR output corresponding to each word blob includes the following steps - (i) Coefficient extraction; (ii) State separation; (iii)Auto correction of the formula unit; and (iv) Auto correction of the entire equation using Context Table.

Table 2.3: Part of the Error list

correct input	all observed outputs given by OCR
g	8 S
O	0
3	8 'E s w
a	3. 21 8 El 8.
l	1 I
s	S

### 2.4.1 Coefficient Extraction

Coefficient extraction is done by matching its regular expression  $[2-9]^+[0-9]^*$  at the beginning of  $S_{chemical}$  as it has numerical values.

### 2.4.2 State separation

There are 4 physical states of a chemical compound which are represented by ‘(s)’, ‘(g)’, ‘(l)’ and ‘(aq)’. To detect the physical state of the compound, regular expression  $[() [A-Za-z]^{1,2}]$  is used and the checking starts from the end of  $S_{chemical}$ . The matched substring ( $S$ ) is extracted from  $S_{chemical}$  and Algorithm 1 is run. In this algorithm,  $S$  and  $H$  are taken as input and all possible *Combinations* of corrected OCR output is produced by *GetAllCombinations* (See Fig. 2.16). These *Combinations* are compared with ‘s’, ‘g’, ‘l’ and ‘aq’. If no match is found, the substring extracted from  $S_{chemical}$  is a radical, not a state; else, we separate the state from the compound.

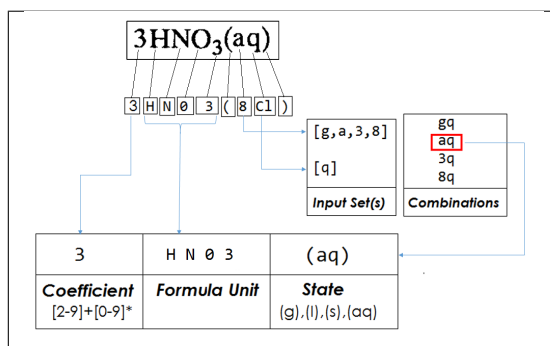


Figure 2.16: Extracting the formula unit, numeric coefficient and physical state from a chemical compound.

### 2.4.3 Auto correction of the formula unit

After extracting coefficient and state, only the formula unit is left in  $S_{chemical}$ . The algorithm for autocorrection of each formula unit is done in two steps (See Algorithm 1 and 2). First, Algorithm 1 is performed on the formula unit. Next, the output of Algorithm 1 (*Combinations*) are matched against a nearly exhaustive list of all molecules, chemical compounds, radicals and atoms namely *ChemList* collected from Wikipedia<sup>1</sup>. If a perfect match is found, that match is considered as the *Corrected* formula unit (See Fig. 2.17). But in case of no match, we go for longest common substring(s) (LCS) match (*SubMatch*). If there is only one *SubMatch* then the corresponding formula unit in *ChemList* is considered as the *Corrected* formula unit; else the *SubMatches* are considered as *PossibleFormulaUnits*.

<sup>1</sup>[http://en.wikipedia.org/wiki/Dictionary\\_of\\_chemical\\_formulas](http://en.wikipedia.org/wiki/Dictionary_of_chemical_formulas)

---

**Algorithm 1** Get All Combinations from the Error Hash Map

---

```

1: procedure GETALLCOMBINATIONS( $S, H$ )
2:   for all  $element(s) \in S$  do
3:      $InputSet(s) \leftarrow H.Get(element)$ 
4:     if  $InputSet(s)$  is NULL then
5:       RETURN ▷ Not in Error Map
6:     else
7:       if  $length(element) = 1$  then
8:          $InputSets = \cup [element]$ 
9:         ▷ Element might be correct output but still in the error list
10:        for other inputs
11:       else
12:         Ignore
13:         ▷ Input is one character, output length > 1 means error
14:       end if
15:     end if
16:   end for
17:    $Combinations \leftarrow CartesianProduct(InputSets)$ 
18:   Return  $Combinations$ 
19: end procedure

```

---

**2.4.4 Auto correction of the entire equation using Context Table**

Here, we have all the possible formula units and try to find out the *FinalEquation* in the context of the equation itself. Algorithm 3 takes all *Corrected* and *PossibleFormulaUnits* and returns the *FinalEquation* by forming the Context Table. Chemical equations have the same periodic elements in the left hand side, called *Reactants* as that in the right hand side, called *Products*. All the periodic elements follow the regular expression  $[A-Z][a-z]^*$ . So, for each *PossibleFormulaUnit*, the set of periodic elements in the *Reactants*,  $P_R$  and in the *Products*,  $P_P$  are computed and stored in the *ContextTable*. When the set difference of  $P_R$  and  $P_P$  in the table is empty, that *PossibleFormulaUnit* is considered as *Corrected* and included in the *FinalEquation* (See Fig. 2.18).

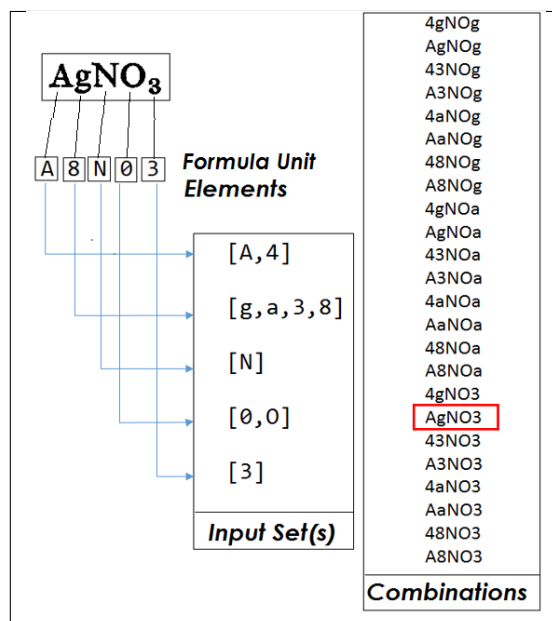


Figure 2.17: Example of autocorrection of a formula unit.

But if the above condition comes true for multiple possibilities, we cannot decide which of the possible formula units are actually in the original equation. The algorithm shows multiple *FinalEquations*. This is considered an *ERROR* case. An example of formation of context table is demonstrated in Fig. 2.18. The co-efficient and state (if any) are added with their corresponding *Corrected* formula unit. The *FinalEquation* is then converted to  $\text{\LaTeX}$  format using *mhchem* package.

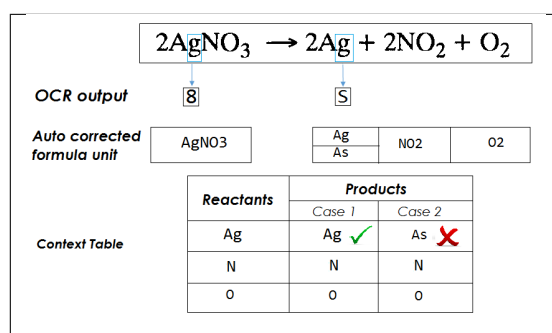


Figure 2.18: Formation of context table.

---

**Algorithm 2** Find Match between ChemList and Combinations derived from Algorithm 1

---

```

1: procedure FINDMATCH(ChemList, Combinations)
2:   for all Combinations do
3:     match with ChemList
4:   end for
5:   if #(Match Found) = 1 then
6:     Corrected  $\leftarrow$  Match
7:     Return Corrected
8:   else if #(Match Found) = 0 then
9:     SubMatch  $\leftarrow$  LCS(Combinations, ChemList)
                                 $\triangleright$  LCS : Longest Common Substring
10:    if NumberOf(SubMatch)=1 then
11:      Corrected  $\leftarrow$  SubMatch
12:      Return Corrected
13:    else
14:      PossibleFormulaUnit(s)  $\leftarrow$  SubMatch
15:      Return PossibleFormulaUnit(s)
16:    end if
17:  else
18:    PossibleFormulaUnits  $\leftarrow$  Matches
19:    Return PossibleFormulaUnit(s)
                                 $\triangleright$  Multiple matches
20:  end if
21: end procedure

```

---

### 2.4.5 Experimental Result

We have implemented our algorithm in MATLAB 8.3.0.532 (R2014a) in a PC (Intel(R) Core(TM) i5-3337U CPU @ 1.80GHz running Windows 8). The proposed method has been tested on a dataset consisting 234 document pages. Out of 234 pages 50 pages are taken from ICDAR 2013 Math-zone segmentation datasets and other document pages are scanned from different Mathematics and Chemistry books. The summary of the experimental results is shown in Table 2.4. Out of 3406 chemical compounds in the test

---

**Algorithm 3** Auto-Correction of the entire equation using chemical context table

---

```

1: procedure GETFINALEQN(PossibleFormulaUnit, Corrected)
2:   Include all Corrected units in FinalEquation
3:   Count  $\leftarrow$  0
4:   for every PossibleFormulaUnit do
5:     compute( $P_R$ )
6:        $\triangleright P_R$  : Set of periodic elements in Reactants
7:     compute( $P_P$ )
8:        $\triangleright P_P$  : Set of periodic elements in Products
9:     if  $P_R - P_P = \emptyset$  then
10:      Corrected  $\leftarrow$  PossibleFormulaUnit
11:      Include that in the FinalEquation
12:      Count  $\leftarrow$  Count + 1
13:    end if
14:  end for
15:  if Count  $\geq$  2 then
16:    Multiple Corrected compounds
17:    Multiple FinalEquations
18:     $\triangleright$  ERROR
19:  end if
20:  Return FinalEquation(s)
21: end procedure

```

---

dataset, 114 were partially corrected and 52 could not be corrected at all. The overall accuracy of autocorrection is 95.12%. These results are quite encouraging. See the sample image (Fig. 2.19(a)). Corresponding segmented displayed chemical equations are shown in Fig. 2.19(b). Fig. 2.19(c) shows the direct OCR output where ‘i’ has been wrongly identified as ‘l’, ‘I’ and ‘1’ (for *Si* in all the lines of Fig. 2.19(c)). Similarly ‘O’ results in ‘0’ (line 1,2,3). ‘S’ sometimes is detected as ‘5’ (line 5). Our auto correction algorithm remedies these issues. Fig. 2.19(d) demonstrates the effect of our auto correction algorithm. This algorithm is targeted towards chemical equation with linear representation. More results are included in the following website <https://sites.google.com/site/chemeqndb/>



<p>(a) Preparation of Amorphous Silicon. This is the common form of silicon and may be obtained by the following methods:</p> <p>(i) By heating silica or well powdered quartz with magnesium powder in a fire clay crucible.</p> $\text{SiO}_2 + 2\text{Mg} \longrightarrow \text{Si} + 2\text{MgO}$ <p>The product is washed first with dilute hydrochloric acid to remove magnesium, recharged with magnesium and magnesium chloride, recharged to the reduction of silica by magnesium.</p> $\text{SiO}_2 + \text{Mg} \longrightarrow \text{Mg}_2\text{Si} + 2\text{MgO}$ <p>It is now washed with hydrofluoric acid to remove any unchanged silica, where silicon is left behind as a grey amorphous powder.</p> <p>(ii) By heating fluorosilicates or silicon tetrafluorides with alkali metals.</p> $\text{K}_2\text{SiF}_6 + 4\text{K} \longrightarrow 4\text{KF} + \text{Si}$ $\text{SnF}_4 + 4\text{K} \longrightarrow 4\text{KF} + \text{Si}$ <p>(iii) By heating a mixture of silica and powdered coke or charcoal in a current of dry chlorine.</p> $\text{SiO}_2 + 2\text{C} + 2\text{Cl}_2 \longrightarrow \text{SiCl}_4 + 2\text{CO} \uparrow$ <p>The vapours of <math>\text{SiCl}_4</math> are then passed over an alkali metal (Na or K).</p> $\text{SiCl}_4 + 4\text{Na} \longrightarrow 4\text{NaCl} + \text{Si}$	<p>(b)</p> $\text{SiO}_2 + 2\text{Mg} \longrightarrow \text{Si} + 2\text{MgO}$ $\text{SiO}_2 + 4\text{Mg} \longrightarrow \text{Mg}_2\text{Si} + 2\text{MgO}$ $\text{K}_2\text{SiF}_6 + 4\text{K} \longrightarrow 4\text{KF} + \text{Si}$ $\text{SnF}_4 + 4\text{K} \longrightarrow 4\text{KF} + \text{Si}$ $\text{SiO}_2 + 2\text{C} + 2\text{Cl}_2 \longrightarrow \text{SiCl}_4 + 2\text{CO} \uparrow$ $\text{SiCl}_4 + 4\text{Na} \longrightarrow 4\text{NaCl} + \text{Si}$
<p>(c)</p> $\text{SiO}_2 + 2\text{Mg} \longrightarrow \text{Si} + 2\text{MgO}$ $\text{SiO}_2 + 4\text{Mg} \longrightarrow \text{Mg}_2\text{Si} + 2\text{MgO}$ $\text{K}_2\text{SiF}_6 + 4\text{K} \longrightarrow 4\text{KF} + \text{Si}$ $\text{SnF}_4 + 4\text{K} \longrightarrow 4\text{KF} + \text{Si}$ $\text{SiO}_2 + 2\text{C} + 2\text{Cl}_2 \longrightarrow 5\text{SiCl}_4 + 2\text{CO} \uparrow$ $\text{SiCl}_4 + 4\text{Na} \longrightarrow 4\text{NaCl} + \text{Si}$	<p>(d)</p> $\text{SiO}_2 + 2\text{Mg} \longrightarrow \text{Si} + 2\text{MgO}$ $\text{SiO}_2 + 4\text{Mg} \longrightarrow \text{Mg}_2\text{Si} + 2\text{MgO}$ $\text{K}_2\text{SiF}_6 + 4\text{K} \longrightarrow 4\text{KF} + \text{Si}$ $\text{SnF}_4 + 4\text{K} \longrightarrow 4\text{KF} + \text{Si}$ $\text{SiO}_2 + 2\text{C} + 2\text{Cl}_2 \longrightarrow \text{SiCl}_4 + 2\text{CO} \uparrow$ $\text{SiCl}_4 + 4\text{Na} \longrightarrow 4\text{NaCl} + \text{Si}$

Figure 2.19: Experimental result; (a) Input image; (b) Segmented Chemical equation; (c) OCR output in PDF format; (d) Output of the proposed method.

Table 2.4: Summary of Experimental Results

# Total Images	234
# Total Chemical Operands	3460
DE segmentation accuracy	95.12%

## 2.4.6 Error Analysis

Here we try to analyse the sources of some of the errors which have negative effect on the performance figures both for segmentation and classification.

### Auto Correction Error

The Auto Correction Algorithm fails when the chemical equation contains some text such as *and*, *or* etc between two displayed equation in the same line. Also, when the chemical compound is written in formats such as  $(\text{Na}_2\text{SiO}_3)_n$ , only  $\text{Na}_2\text{SiO}_3$  is detected based on *ChemList*. Some equations have conditions (pressure, temperature) written over the arrows. We have not ventured in that yet. But the error case that has been mentioned in Algorithm 3 has very less probability of occurrence. Hence it is ignored.

## Chapter 3

## Conclusion

We have presented an automated chemical equation segmentation and chemical context based auto correction system that is able to provide the exact searchable format of linear chemical equations in any document image. The method is based on detecting operators as  $+$ ,  $-$  and  $\rightarrow$  which are common in both chemical and non-chemical equations, segmenting the displayed equations and then running them through an open source OCR to classify. A publicly available database for mathematical documents and scanned images of various chemistry books in both English and Bengali language are used in this study. The experimental results demonstrate the efficiency of our proposed method. The present work points to several new research avenues to be explored further. The over-all performance itself demands further research in this area. Excessive degradation due to aging and improper digitization of the document images give rise to broken and merged characters which give conflicting output from the OCR. In the future, design of a better integrated OCR specifically for chemical equations would be ventured in. In addition, formation of “electron bond matrix” of a chemical compound in a reaction mentioned in the introduction chapter would be a high-level goal in the future. Auto correction on non-linear or bond structure representations of chemical equations could be ventured in.

# Bibliography

- [1] D. Blostein and A. Grabavec. “*Recognition of Mathematical Notation*”, *Handbook of Character Recognition and document Image Analysis*, 557–582, 1997.
- [2] K-F. Chan and D-Y. Yeung. “*Mathematical Expression Recognition: A Survey*”. *IJDAR*, Vol. 3, no. 1, 3–15, 2000.
- [3] U. Garain and B. B. Chaudhuri. *On OCR of Printed mathematical Expressions*. “*Digital Document Processing*”, Ed. B. B. Chaudhuri, *Advances in pattern Recognition*, 235–259 , 2007.
- [4] R. Fateman, T. Tokuyasu, B. Berman, and N. Mitchell. “*Optical character recognition and parsing of typeset mathematics*”, *Visual Commun. And Image Representation*, Vol 7, no 1, 2–15, 1996.
- [5] J. Y. Toumit, S. Garcia-Salicetti, and H. Emptoz. “*A hierarchical and recursive model of mathematical expressions for automatic reading of mathematical documents*”. In *Proc. of ICDAR*, 116–122, 1999.
- [6] A. Kacem, A. Beliad and M. Ben Ahmed. “*Automated Extraction of printed mathematical formulas using fuzzy logic and propagation of context*”, *IJDAR*, vol.4 no. 2, 97–108, 2001.
- [7] M. Suzuki, F. Tamari, R. Fukuda, S. Uchida and T. Kanahori. “*INFTY - An Integrated OCR system for Mathematical Documents*”, *Proc. of ACM Symposium on Document Engineering*, 95–104, 2003.
- [8] Utpal Garain. “*Identification of Mathematical Expressions in Document Images*”, *Proc. of ICDAR*, 1340–1344, 2009.
- [9] Utpal Garain. “*Recognition of Printed Handwritten Mathematical Expressions*”, *Ph.D Thesis, ISI, Kolkata, India*, 2005.

- [10] B. B. Chaudhuri and U. Garain. “*Extraction of type atyle based meta-information from Imaged documents*”, *IJDAR*, vol. 3 no. 3, 138–149, 2001.
- [11] J. Jin, X. Han and Q. Wang. “*Mathematical formulas extraction*”, *Proc. of ICDAR*, 1138–1141, 2003.
- [12] D. M. Drake and H. S. Baird. “*Distinguishing mathematical notation from english text using computational geometry*”, *Proc. of ICDAR*, 1270–1274, 2005.
- [13] Y.-S. Guo, L. Huang and C.-P. Liu. “*A new approach for understanding of structure of printed mathematical expressions*”, *Proc. of ICMLC*, 2633–2638, 2007.
- [14] We-Te Chu and Fan liu. “*Mathematical formula detection from heterogeneous document Images*”, *Proc. of CTAAI*, 140–146, 2013.
- [15] S. P. Chowdhury, S. Mandal, A. K. Das, and B. Chanda. “*Segmentation of Text and Graphics from Document Images*”, in *Proc. of ICDAR*, 619–623, 2007.
- [16] R. C. Gonzalez and R. Woods. “*Digital Image Processing*”. Addison-Wesley, 1992.
- [17] S. Mandal, S. P. Chowdhury, A. K. Das, and B. Chanda, “*A simple and effective table detection system from Document Images*”, in *Proc. of IJDAR*, Vol. 8(2), 172–182, 2006.
- [18] S. P. Chowdhury, S. Mandal, A. K. Das, and B. Chanda, *Segmentation of Text and Graphics from Document Images*, In *Proc. of ICDAR*, pp. 619–623, 2007.
- [19] S. Mandal, S. P. Chowdhury, A. K. Das, and B. Chanda, *A simple and effective table detection system from Document Images*, *IJDAR*, Vol. 8(2), 172–182, 2006.
- [20] D. Blostein and A. Grabavec, *Recognition of Mathematical Notation*, *Handbook of Character Recognition and document Image Analysis*, 577–582, 1997.
- [21] K-F. Chan and D-Y. Yeung, *Mathematical Expression Recognition: A Survey*, *IJDAR*, Vol. 3, no: 1, 3–15, 2000.

- [22] U. Garain and B. B. Chaudhuri *An OCR of Printed mathematical Expressions*, *Digital Document Processing*, Ed: B. B. Chaudhuri, Advances in pattern Recognition, 235–259 , 2007.
- [23] A. Fujiyoshi, M. Suzuki, S. Uchid, *Grammatical Verification for Mathematical Formula Recognition Based on Context-Free Tree Grammar*, *Mathematics in Computer Science*, 279–298, 2010.
- [24] M. E. Algorri, M. Zimmermann, C. M. Friedrich, S. Akle, and M. Hofmann-Apitius, *Reconstruction of chemical molecules from images*, In Proc. 29th Annual International IEEE Conference on Engineering in Medicine and Biology Society, 4609–4612, 2007.
- [25] M. E. Algorri, M. Zimmermann, and M. Hofmann-Apitius, *Automatic recognition of chemical images*, In pro. Eighth Mexican International Conference on Current Trends in Computer Science, 41–46, 2007.
- [26] J. Park, G. R. Rosania, K. A. Shedden, M. Nguyen, N. Lyu, and K. Saitou, *Automated extraction of chemical structure information from digital raster images*, *Chemistry Central journal*, vol. 3(1), 2009.