

Automated Generation of Searchable PDF Format of Chemical Equations from Document Images

Prerana Jana, Anubhab Majumdar, Sekhar Mandal

Department of Computer Science and Technology
Indian Institute of Engineering Science and Technology Shibpur, India
Email: (prerana.jana, anubhabmajumdar93@gmail.com, sekhar@cs.becs.ac.in)

Bhabatosh Chanda

Electronics and Communication Sciences Unit
Indian Statistical Institute, Kolkata, India
E-mail : chanda@isical.ac.in

Abstract—The abstract goes here.

I. INTRODUCTION

Here goes the introduction and previous work

II. PROPOSED WORK

Major steps involved in the automated generation of searchable PDF format are given below.

- Segmentation of displayed chemical equation
- Recognition of various chemical symbols present in chemical equations
- Optical character recognition of each reactant
- Auto correction of reactants and the chemical equation itself
- Generation of chemical equation in searchable PDF format

The details of the above steps are given in the following subsections.

A. Segmentation of displayed chemical equation

Segmentation of displayed equations and then extracting only the chemical equations from a document image has been done using the methodology proposed in [1]. Some improvements on the segmentation error has been attempted. For the reactants over the arrow, the main problem is single character extraction. In [1], component count has been done in the bounding box of one blob. Here, if we find multiple components, each component's bounding box is calculated. If one of the component's bounding box is equal to the bounding box of its blob, we remove the components with smaller bounding boxes (See Fig.??). This is based on the positional context that an arrow with or without reactants over or below it will have the same bounding box as that of its word blob.

B. Recognition of various chemical symbols present in chemical equations

A chemical equation is a way of representing a chemical reaction in symbolic form. Chemical equations consists of reactants separated by myriad chemical symbols. The symbols along with their significance are listed below.

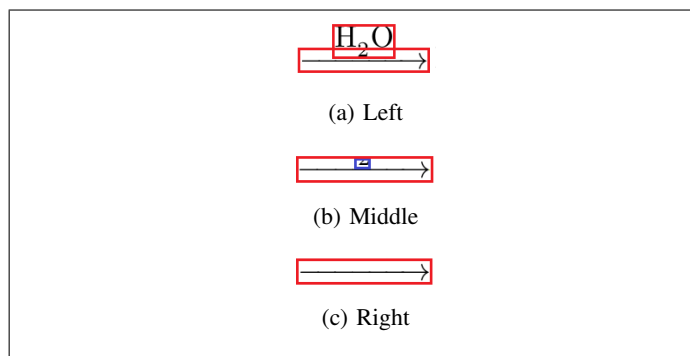


Fig. 1: (a). Arrow with reactants over and below it. (b). Bounding Box of the arrow. (c). After removal of smaller components

- + : Separate the reactants
- \rightarrow , \leftarrow : Separate the reactants from products in irreversible reactions; also denote the direction of reaction
- \leftrightarrow , \rightleftharpoons : Separate the reactants from products in reversible reactions
- = : Shows stoichiometric equality in chemical equations
- \uparrow : Used to denote gaseous compound
- \downarrow : Used to denote sediments formed after a reaction

We begin with the extracted displayed chemical equation (DCE) from the step above and run a HRLS algorithm. This results in the coalescing of the chemical compounds into a word blob as shown in Fig.2b.

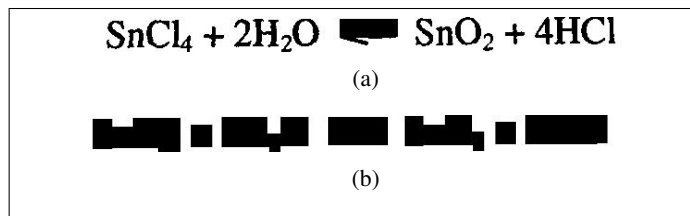


Fig. 2: (a). Original DCE. (b). DCE after closing operation.

All the single components are identified and segregated from the original equation and classified using a decision tree shown in Fig.3.

The function of each node of the decision tree is detailed below.

- **#of component** : This module count number of disjoint components in the input symbol.
- **Crossing** : We measures the number of transitions from object to background pixel or vice versa for each column while moving along the rows and consider the maximum value.
- **OCR** : The input symbol is run through OCR to identify positively the + symbol. Other symbols return erroneous result.
- **Aspect Ratio** : Calculates the $\frac{height}{width}$ ratio of input image.
- **Distance Transform** : The input symbols are closed with a disc structuring element of radius $\frac{m}{2}$ where m is $\min(height, width)$ of the input symbol. The distance transform is applied and the location of the global maxima denotes the arrow head; hence determines the direction of the arrow.

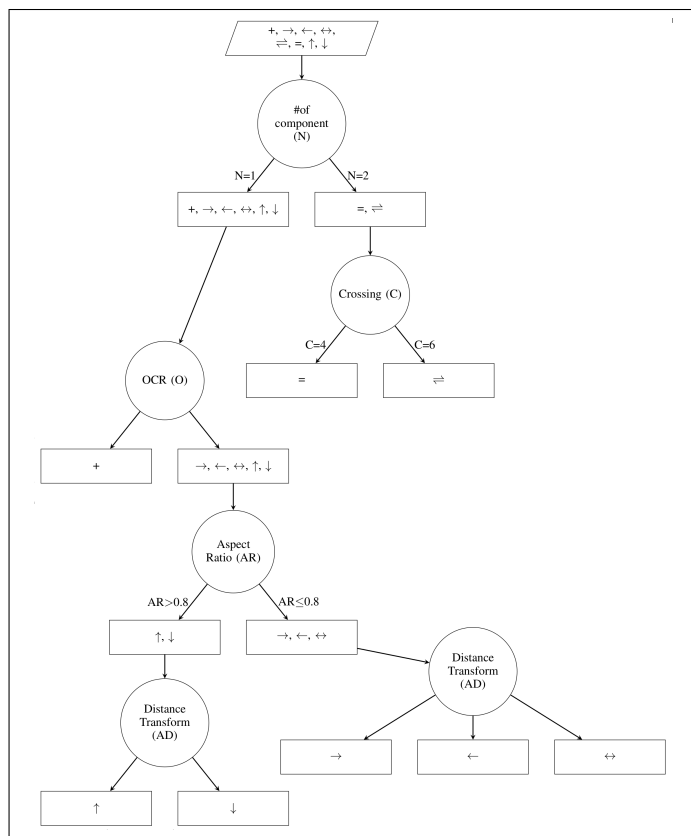


Fig. 3: Decision Tree for Chemical Symbol Classification

Certain chemical element like Carbon, Sulfur etc.(see Fig.4) may also be input to the tree and they get erroneously identified as \uparrow or \downarrow because characters have $AR > 0.8$.

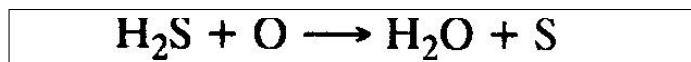


Fig. 4: DCE having element represented by single character

To fix this, we match the symbols classified with the original equation blob image. For each $+$, \rightarrow , \leftarrow , \leftrightarrow , \rightleftharpoons and $=$ in the classified symbol set, we check its immediate right blob. If the blob has single component in the original equation image, then it must be an single character element because these symbols must be followed by reactants. Also the first blob is checked to see if its a single character element or not. Thus the chemical symbols are successfully segregated and classified from the chemical equation.

C. Optical character recognition of each reactant

Each segment of a displayed chemical equation is divided into three zones; namely upper zone, middle zone and lower zone(see Fig.5). To identify the three zones of a DCE zone, uppermost and lowermost co-ordinates of each connected component below the same DCE zone are also obtained. The median of uppermost coordinate, and median of lowermost co-ordinate of such components in DCE 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 DCE zone. Similarly, the median of uppermost co-ordinate of the components in the DCE zone generates a horizontal line. This horizontal line, called top line, separates the middle and upper zones of the DCE zone. The subscripts in a DCE 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 DCE zone we have detected the subscripts and superscripts. Each reactant

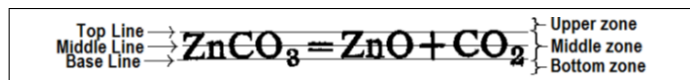


Fig. 5: Presence of different zones in DCE

form one word blob. Each character forming one word blob is run through an OCR engine, Matlab OCR. Each character image is returned by OCR engine in corresponding text format and saved in a string with a space split, $S_{reactant}$ for further processing. For each superscript and subscript, '^' and '_' are inserted before them respectively in $S_{reactant}$ for denoting purposes.

D. Auto correction of reactants in chemical equations

Matlab OCR does not give perfect result for chemical equations which has subscripts and superscripts as well. Hence, to generate the exact equation from the image, auto correction is required in chemical database. First an error map is created based upon our observation on multiple databases (see Table I). Next this table is stored into a hash map where the key is the OCR output and its corresponding value is possible input set. Auto correction is performed based on this error hash map, H .

Each chemical reactant in any equation has the following format- $[\text{Coefficient}]^{[0,1]}[\text{Chemical compound}][\text{state}]^{[0,1]}$. Coefficient has numerical values, its regular expression is taken as $[2-9]+[0-9]^*$. The first digit will not be 0 as coefficient cannot start with that. There are 4 states of a chemical compound which are represented by '(s)', '(g)', '(l)' and '(aq)'. To detect if the compound has any state representation, the following

TABLE I: Error list

correct input	all observed outputs given by OCR
g	8 3
n	11 1'1 11 11 X1 11
O	0
3	8 'E s w
a	3. 3 21 8 El 8.
q	Cl Q
H	1-1 1-1 1-1 1-1
2	7 4 Z z
l	1 I
I	1
u	11 U 11 11 11
i	1 1 1
5	'S
4	A
Z	2
r	1
e	C
s	S

regular expression $[(\text{[a-z]}^{1,2})]$ is checked in the substring of length 4 extracted from the end of $S_{reactant}$ (blank spaces are ignored here). If there is a match, we run the autocorrection algorithm on it and compare the results with 's', 'g', 'l' and 'aq'. If no match is found, the substring extracted from $S_{reactant}$ is a radical, not a state else, we remove the state from the reactant. After extracting coefficient and state, only chemical compound, CC is left in $S_{reactant}$. So, CC is a set containing OCR output of each character of the chemical compound.

The algorithm for autocorrection of each chemical compound is given below. In first step, CC and error map, H are taken as input and all possible combinations of corrected OCR output is produced by *GetAllCombinations*. Next, these *Combinations* are matched against a nearly exhaustive list of all molecules, chemical compounds, radicals and atoms named *ChemList*. If a perfect match is found, that is considered as the *Corrected* chemical compound for that CC . But in case of no match, longest common substring(s) (LCS) between *Combinations* and *ChemList* are extracted and considered as *SubMatch*. If number of such *SubMatch*(es) is 1 that *SubMatch* is considered as the *Corrected* chemical compound for that CC ; else they are considered as *PossibleReactants*. All the *Corrected* compounds are included in the *FinalEquation*.

Here we have all the possible reactants and try to find out the *FinalEquation* in the context of the equation itself. Chemical equations have the same elements in the left hand side (LHS) as that in the right hand side (RHS). All the periodic elements follow the regular expression $[\text{A-Z}][\text{a-z}]^*$. So, for each possibility, the set of periodic elements in the left side of the equation, P_{LHS} and in the right side, P_{RHS} is computed. When the intersection of these two sets are empty, that *PossibleReactant* is considered as *Corrected* and included in the *FinalEquation*. But if the above condition comes true for multiple possibilities, we cannot decide which of the possible reactants are actually in the original equation. The algorithm shows multiple *FinalEquations*. This is considered as an *ERROR* case.

Algorithm 1 Auto-Correction of each reactant

```

1: procedure GETALLCOMBINATIONS( $CC, H$ )
2:   for all  $element(s) \in CC$  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:       end if
10:      end if
11:    end for
12:     $Combinations \leftarrow CartesianProduct(InputSets)$ 
13:  end procedure

14: procedure FINDMATCH( $ChemList, Combinations$ )
15:   for all  $Combinations$  do
16:     match with  $ChemList$ 
17:   end for
18:   if Match Found then
19:      $Corrected = Match$ 
20:   else
21:      $SubMatch \leftarrow LCS(Combinations, ChemList)$ 
22:     ▷ LCS : Longest Common Substring
23:     if NumberOf( $SubMatch$ )=1 then
24:        $Corrected = SubMatch$ 
25:     else
26:        $PossibleReactants \leftarrow SubMatch$ 
27:     end if
28:   end if
29: end procedure

```

Algorithm 2 Auto-Correction of equation

```

1: procedure GETFINALEQUATION( $PossibleReactants$ )
2:   for every possibility do
3:     compute( $P_{LHS}$ )
4:     ▷  $P_{LHS}$  : Set of periodic elements in the L.H.S
5:     compute( $P_{RHS}$ )
6:     ▷  $P_{RHS}$  : Set of periodic elements in the R.H.S
7:     if  $P_{LHS} \cap P_{RHS} = \emptyset$  then
8:        $Corrected \leftarrow PossibleReactant$ 
9:       Include that in the FinalEquation
10:    end if
11:  end for
12:  if multiple possibilities get  $P_{LHS} \cap P_{RHS} = \emptyset$  then
13:    Multiple Corrected compounds
14:    Multiple FinalEquations
15:    ▷ ERROR
16:  end if
17: end procedure

```

E. Generation of ground truth data in PDF format

III. EXPERIMENTAL RESULT

Here goes the results.

IV. CONCLUSION

The conclusion goes here.

REFERENCES

- [1] H. Kopka and P. W. Daly, *A Guide to L^AT_EX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.