Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

ARTICLE TYPE

A Neural Approach for Detecting Inline Mathematical Expressions from Scientific Documents

Sreekanth Madisetty*[1] | Kaushal Kumar Maurya[1] | Akiko Aizawa[2] | Maunendra Sankar Desarkar[1]

[1]Indian Institute of Technology Hyderabad, Telangana, India
[2]University of Tokyo, National Institute of Informatics, Tokyo, Japan

Correspondence
*Corresponding author name. Email: cs15resch11006@iith.ac.in

Summary

Scientific documents generally contain multiple mathematical expressions in them. Detecting inline mathematical expressions is one of the most important and challenging tasks in scientific text mining. Recent works that detect inline mathematical detection in scientific documents have looked at the problem from an image processing perspective. There is little work that has targeted the problem from NLP perspective. Towards this, we define a few features and applied Conditional Random Fields (CRF) to detect inline mathematical expressions in scientific documents. Apart from this feature based approach, we also propose a hybrid algorithm that combines Bidirectional Long Short Term Memory networks (Bi-LSTM) and feature-based approach for this task. Experimental results suggest that this proposed hybrid method outperforms several baselines in the literature and also individual methods in the hybrid approach.

KEYWORDS:
Inline mathematical expression, Independent mathematical expression, scientific text mining, Deep Learning, Natural Language Processing

1 | INTRODUCTION

Almost every scientific research paper is mathematically rich. Similarly, mathematical expressions are very common in educational books and are often seen in documents related to policy decisions, regulations etc. Automated understanding, indexing or retrievals of information from such kinds of documents require proper identification and extraction of these mathematical expressions.

Recent advances in Information Retrieval (IR) and Natural Language Processing (NLP) have enabled researchers to automatically extract and use scientific information of various forms from scientific documents. Metadata (title, authors, abstract, etc.) can be extracted from PDF documents by applying machine learning algorithms. PDF structure information, figures or tables from scientific documents can also be extracted. Although extracting text related content from PDF documents is easy, extracting non-text (math) related content is a challenging task. It is not straightforward to extract mathematical expressions because they follow different styles and patterns, they can be present inside a regular sentence, inside tables, their syntactic appearance can be similar to that of regular words or symbols.

Mathematical expressions in scientific documents can be classified into two categories: a) Isolated or Independent mathematical expressions; b) Inline mathematical expressions. Isolated mathematical expressions are those expressions that are written outside of the text region and have some layout or style in the scientific documents. So, it is easy to identify them by applying some Optical Character Recognition (OCR) techniques with predefined features. Inline mathematical expressions are mixed with normal text and there is no layout or pattern for inline mathematical expressions. It is very difficult and challenging to identify these types of expressions. Example inline and isolated mathematical expressions from the scientific document are shown in Figure 1.

Inline mathematical expression detection can be helpful in many ways. In Information Retrieval (IR), it can be helpful in better indexing and retrieval tasks. It also reduces the errors in sentence parsing. Also, it further advances the scientific text mining as mathematical concepts often convey important information.

In this paper, we propose a hybrid approach that combines both feature based (CRF) and neural network based (Bidirectional Long Short Term Memory network (Bi-LSTM)) to detect inline mathematical expressions in scientific documents. We adopt the dataset from (Iwatsuki, Sagara, Hara, and Aizawa (2017)) for our task. Figure 2 shows the flow diagram of our proposed approach. First, PDF scientific documents are converted into corresponding *xhtml* files. These *xhtml* files contain multiple features related to individual tokens extracted from the scientific documents, and also information about the document itself. Few important token level features extracted from the scientific documents are bounding boxes, font type, font id, class label for inline math expression, word id, sentence id, section name, etc. Bi-LSTM model makes use of extracted tokens and CRF model makes use of extracted features from *xhtml* files. From extracted tokens and extracted features, our proposed work attempts to identify whether it is a part of an inline mathematical expression or not.

Taking SAN as an example, the multi-step reasoning module computes over T memory steps and outputs the answer span. It is based on an RNN, similar to IRN in Fig. 3.5. It maintains a state vector, which is updated on each step. At the beginning, the initial state $\mathbf{s}_0$ is the summarized question vector computed by Eqn. 3.6. At time step t in the range of $\{1, 2, ..., T\}$, the state is defined by $\mathbf{s}_t = \text{RNN}(\mathbf{s}_{g-1}, \mathbf{x}_t)$, where $\mathbf{x}_t$ contains retrieved information from memory using the previous state vector as a query via the attention process: $\mathbf{M}$: $\mathbf{x}_t = \sum_j \gamma_j \mathbf{m}_j$ and $\gamma = \text{softmax}(\mathbf{s}_{t-1}{}^\top \mathbf{W}^{(att)}\mathbf{M})$, where $\mathbf{W}^{(att)}$ is a trainable weight matrix. Finally, a bilinear function is used to find the start and end point of answer spans at each reasoning step t, similar to Eqn. 3.7 and 3.8, as

$$\mathbf{p}_t^{(start)} = \text{softmax}(\mathbf{s}_t{}^\top \mathbf{W}^{(start)}\mathbf{M}), \quad (3.9)$$

$$\mathbf{p}_t^{(end)} = \text{softmax}([\mathbf{s}_t; \sum_j p_{t,j}^{(start)} \mathbf{m}_j]^\top \mathbf{W}^{(end)}\mathbf{M}), \quad (3.10)$$

where $p_{t,j}^{(start)}$ is the j-th value of the vector $\mathbf{p}_t^{(start)}$, indicating the probability of the j-th passage word being the start of the answer span at reasoning step t.

FIGURE 1 Example of in-line and independent mathematical expressions from a scientific document (Gao et al. (2019)) (blue rectangle- inline math expression and red rectangle - isolated or independent math expression)

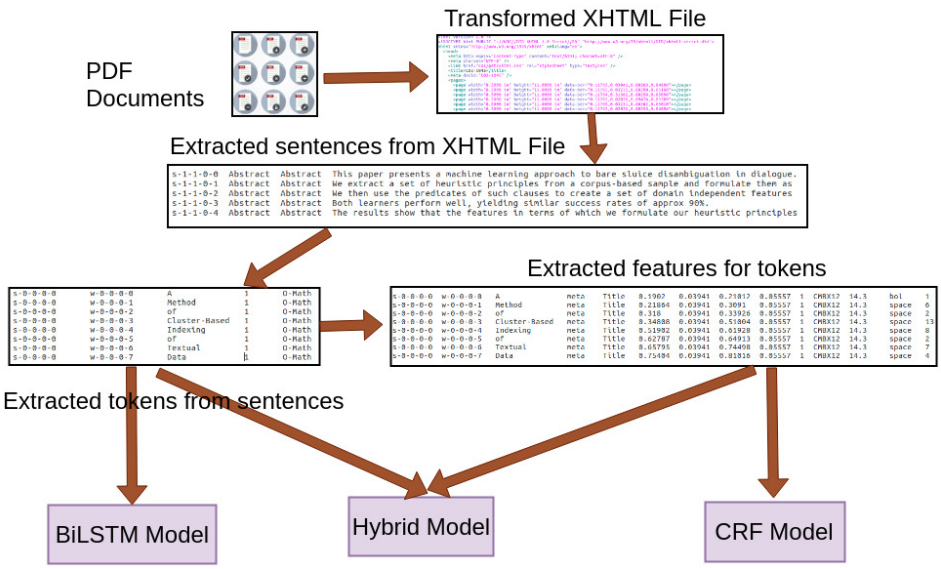

FIGURE 2 Flow diagram of the proposed approach for identification of in-line math expression from scientific PDF document

2 | RELATED WORK

Exploration and analysis of scientific content open many research possibilities like task-oriented content extraction, content summarization, content semantic interpretation, development of knowledge/citation graph, etc. We divide the literature work into two areas: mathematical formula identification and PDF document analysis.

2.1 | Mathematical Formula Identification

A method for processing mathematical formulas in printed documents is presented in (Garain and Chaudhuri (2000)). Mathematical expressions are detected by exploiting some structural properties and recognizing a few most common symbols. An approach is proposed to extract mathematical formulas from document images in (Jin, Han, and Wang (2003)). Isolated formulas are identified by using a parzen window and embedded formulas are detected by using *2-D* structures detection. Features used in this method are line height, space above the line, space below the line, left indent, right indent, the distance between formula and its sequence number. A method to identify embedded mathematical expressions in scanned documents is presented in (Garain, Chaudhuri, and Chaudhuri (2004)). Word n-grams are used as features in this method. (Drake and Baird (2005)) proposed a method to distinguish mathematical notation from natural language using computational geometry. Input to this method is a neighbor graph extracted from image of an isolated textline. The authors classify each vertex and each edge of the neighbor graph separately as math or non-math. A method is developed to identify mathematical expressions in document images in (Garain (2009)). Linguistic information features, shape features, low level image features, etc. are used in this method. (Lin, Gao, Tang, Lin, and Hu (2011)) proposed an approach to identify mathematical formulas in PDF documents. Both embedded and isolated mathematical expressions are identified using rule-based and learning-based methods. Geometric layout features, context content features, character features, etc. are the features used in this method. The same authors also proposed an SVM-based model to detect embedded mathematical expressions in PDF documents (Lin, Gao, Tang, Hu, and Lin (2012)). In this method, they first segment text lines into words and label each word as math formula or normal text. A tool, *Maxtract* is used to analyze and convert scientific documents in (Alkalai, Baker, Sorge, and Lin (2013)). Machine learning based formula identification and statistical line detection are the two segmentation techniques used in this analysis. By using these two techniques the authors showed improvement in the quality of both Maxtract document analysis and its output.

An approach is developed for mathematical formula identification in heterogeneous document images (Chu and Liu (2013)). Here, heterogeneous document images means it may contain tables, figures, text, and math formulas. Features used in this method are density, height of text line, left indent, right indent, and centroid fluctuation. (Tian, Yu, and Sun (2014)) proposed a method to analyze images in PDF documents for mathematical expression retrieval. The modules included in this method are PDF image extraction, graying, binarization, denoising, skew correction, and layout parameter detection. An approach to recognize mathematical formulas in degraded Chinese documents is developed in (Liu et al. (2017)). Convolutional neural networks (CNN) is used to split and recognize math formulas. They applied the method to the dataset containing only 100 images. A method to detect inline mathematical expressions in scientific documents is proposed in (Iwatsuki et al. (2017). A sequence of text is given as input to the model. Features used are layout features (font types,font size, etc.)and linguistic features (context *n-grams*). A simple Conditional Random Fields (CRF) model is applied using these features.

A tree-based BiLSTM architecture is developed to recognize the online handwritten mathematical expressions in (T. Zhang, Mouchère, and Viard-Gaudin (2018)). A stroke label graph is generated to describe the mathematical expression. Another method to detect handwritten mathematical expressions using convolution neural networks is presented in (Tran, Huynh, Le, Phan, and Bui (2018)). (J. Zhang, Du, and Dai (2018a)) use densely connected convolution neural networks to detect handwritten mathematical expressions. The authors also presented a multi-scale attention model to identify math symbols in different scales. A TAP (Track, Attend and Parse) model to recognize online handwritten mathematical expressions in (J. Zhang, Du, and Dai (2018b)). The authors used tracker and parser for the recognition of mathematical expressions. The tracker use a stack of bidirectional RNNs with GRUs to model the handwritten traces. Next, parser uses GRU with guided hybrid attention to produce LATEX notations.

A method for mathematical variable detection is presented in (Phong, Hoang, Le, and Aizawa (2019)). The authors developed several rules for detection. They also used pre-trained deep convolution neural networks to improve accuracy. They have applied the method to English and Vietnamese documents. The same authors proposed a method for mathematical variable detection using convolution neural networks and SVM in (Phong, Hoang, and Le (2019)). A hybrid feature extraction approach to detect offline handwritten math symbols is discussed in (Ramírez-Piña, Sánchez, Valdovinos-Rosas, and Hernández-Servín (2018)). The authors used wavelet and zoning techniques to define the feature vectors. The method depends on the geometrical and statistical characteristics of symbol images. The authors in (Ohyama, Suzuki, and Uchida (2019)) used U-Net framework (Ronneberger, Fischer, and Brox (2015)) to detect mathematical equations in scientific document images. A method Tangent-CFT for embedding mathematical formulas is presented in (Mansouri et al. (2019)). The authors used two hierarchical representations: Symbol Layout Trees and Operator Trees.

2.2 | PDF Document Analysis

An approach to extract and classify diagrams in PDF documents is proposed in (Futrelle, Shao, Cieslik, and Grimes (2003)). Features used in this method are: number of rectangles, number of data points, number of lines or curves, number of categorical labels, number of short horizontal lines, and number of short vertical lines. Support Vector Machine (SVM) is used in this method. (Lopez (2009)) developed a method to extract bibliographic information from scholarly articles. (Xu, Tang, Tao, Li, and Shi (2013)) presented a method for graph-based layout analysis for PDF documents. Geometric layout features, character features, and image features are used for text segments and non-text segments in this method. A tool, *PDFX*, is developed for conversion of PDF documents to XML documents in (Constantin, Pettifer, and Voronkov (2013)). It is based on rule-based method. Rule sets depend on parameters derived from font and layout characteristics of each article. (Lipinski, Yao, Breitinger, Beel, and Gipp (2013)) analyses different approaches and tools which extract header metadata information from scientific PDF documents. In their evaluation GROBID (Lopez (2009)) peformed the best compared with other methods for arxiv collection. Another tool *CERMINE* (Tkaczyk, Szostek, Fedoryszak, Dendek, and Bolikowski (2015)) is developed for automatic extraction of structured metadata from scientific documents. Features used in this method are: geometric, lexical, sequential, formatting, and heuristics based features. One more tool, *OCR++* (Singh et al. (2016)) is developed for information extraction of scholarly articles. Metadata information, structure information, and bibliography information is extracted using this tool.

2.3 | Sequence Tagging

As discussed earlier, there are very few works in literature for identifying mathematical expressions from scientific documents that accept the input as a sequence of text, as opposed to an image. If the input is considered as a sequence of text, then this identification can also be considered as a sequence labeling task. Recently, deep learning models have been shown good performance in sequence labelling tasks (Huang, Xu, and Yu (2015), Seo, Kembhavi, Farhadi, and Hajishirzi (2016), Ma and Hovy (2016), Akbik, Blythe, and Vollgraf (2018),Alzaidy, Caragea, and Giles (2019),). Here we discuss recent advancements in sequence labeling algorithms that can be helpful to address the problem of mathematical expression detection.

Different LSTMs have been explored for sequence tagging in (Huang et al. (2015)). These models include LSTM, Bi-LSTM, LSTM-CRF, and Bi-LSTM-CRF. The authors have shown that Bi-LSTM-CRF performs better than other models. The same Bi-LSTM-CRF model is used in (Alzaidy et al. (2019)) to extract keyphrases in scientific documents. Here Bi-LSTM and CRF layers are stacked in the neural architecture that is trained and tested for the keyphrase extraction data. Sequence labeling is done by combining Bi-LSTM, CNN, and CRF in (Ma and Hovy (2016)). The authors make use of both word level and character level representation for better performance. Part-of-Speech (POS) tagging and Named Entity Recognition (NER) are the two tasks considered. A method for machine comprehension using Bi-Direction Attention Flow (BiDAF) is presented in (Seo et al. (2016)). This hierarchical process consists of six layers: character embedding layer, word embedding layer, contextual embedding layer, attention flow layer, modeling layer, and output layer. This method can be applied to sequence tagging also.

A framework for multi-task domain adaptation for sequence tagging is presented in (Peng and Dredze (2016)). The method is composed of the learner which can learn shared representations for all datasets, domain projection layer, and a task specific learner to learn a set of feature weights per task.

3 | METHODOLOGY

In this section, we describe our proposed approaches for detection of inline mathematical expressions from scientific documents. We defined the task of inline mathematical expression as:

> **Problem Definition:** For an input text S given as a sequence of tokens $S =< t_1, t_2, \cdots, t_n >$, assign a label to each of the t_i's (for $i \in \{1, \cdots, n\}$. Each t_i is a token from the text, and has additional information available with it like font details, bounding box etc. The labels in this context are *B-Math*, *I-Math*, and *O-Math*. If a token is labeled with B-Math, then it indicates that an inline mathematical expression starts from this token. Similarly, if the label assigned to t_i is I-Math (or O-Math), then it indicates that t_i is inside a mathematical expression (or outside - not part of any mathematical expression).

Given a training data for the above problem, we want to develop supervised machine learning based models to determine the labels associated with the tokens. In this paper, we propose three different methods for this task: (a) a feature-based sequence tagging method, (b) a Bidirectional Long Short Term Memory Networks (Bi-LSTM) based Inline Mathematical Expression Detector (BIMExD) and (c) a Hybrid Inline Mathematical Expression Detector (HIMExD) approach that augments Bi-LSTM network with explicit feature information about the text. We explain these methods in detail in the following subsections.

3.1 | Feature Based Approach

The atomic unit of any textual document, including scientific documents, is a word or token[1]. In our first approach, we identify a set of features for each of the tokens. The set of features used in our feature based method are listed in Table 1. The identified features are good indicators to detect inline mathematical expressions. Once the data is prepared we applied a well known sequence tagging model, Conditional Random Forest (CRF) for detection of mathematical expressions. We call this method as CRF2.

TABLE 1 Features used in our Feature based and Hybrid methods.

S.No	Feature	Type	Description	S.No	Feature	Type	Description
1	token	String	word, math symbol, punctuation	10	sameFoT	Boolean	is same fontType of token used in an independent formula?
2	fontType	String	font type of the token	11	sameFT	Boolean	feature-9 **AND** feature-10
3	fontSize	Numeric	font size of the token	12	single	Boolean	is length of the token is one?
4	fontID	Numeric	font ID (type and size) of the token	13	mainFont	Boolean	fontID(token) == fontID(body text)?
5	SecName	String	section name containing the token	14	url	Boolean	is the token url or part of url?
6	boxName	String	label of block having token	15	mathSymbol	Boolean	does the token consist of math symbol?
7	X1,X2,Y1,Y2	Numeric	bounding-box co-ordinates of token	16	geekSymbol	Boolean	does the token consist of math geek symbol?
8	length	Numeric	length of the token	17	onlyLetter	Boolean	does the token consist of only letters?
9	sameT	Boolean	is same fontID of token used in an independent formula?				

3.2 | Neural Network Based Approach

Recently, neural networks have been shown to achieve very good performance on various NLP tasks such as classification, Named Entity Recognition (NER), text summarization, etc. Recurrent neural networks (RNN) have been used for many of the applications mentioned above. RNN sees current input a_t and hidden state p_{t-1} to predict next state. RNN can not handle long-term dependencies due to vanishing gradient problem. To overcome the drawbacks of RNN, Long Short Term Memory networks (LSTM) are introduced. We propose a Bi-LSTM based Inline Mathematical Expression Detector (BIMExD) to detect inline mathematical expressions in scientific documents. BIMExD architecture is shown in Figure 3. The architecture has two Bi-LSTM layers, a linear layer, and a softmax output layer. The sentence in scientific document is given as input to the model. One-hot encoding of the sentence tokens are given to the first Bi-LSTM layer. The output of this Bi-LSTM layer is given to another Bi-LSTM layer to get the hidden representation of the individual tokens. Bi-LSTM uses forward LSTM for modeling one sentence in the same order as it appears and backward LSTM for modeling sentence in the reverse order. The output of second Bi-LSTM layer is given to linear layer. Finally, softmax is applied in output layer to get the best tags (O-Math, B-Math, I-Math) for the given tokens in the sentence.

3.3 | Hybrid Approach

We propose a Hybrid Inline Mathematical Expression Detector (HIMExD) which combines feature based approach (features as described in Section 3.1) and neural network based approach (as described in Section 3.2). Block diagram of our proposed approach HIMExD is shown in Figure 4.

For a given token t_{ki} from sentence $s_k \subset \mathcal{S}$, a set of 17 features (as described in Table 1) i.e., $F_{ki} = (f_{ki1}, f_{ki2}, \ldots, f_{ki17})$ are extracted. Additionally, each token is also fed through embedding layer to obtain its embedding representation (i.e., $E_{t_{ki}}$). $E_{t_{ki}}$ is then passed through a hidden BiLSTM layer (as described in section 3.2) to get its intermediate representation $h_{t_{ki}} = [\overrightarrow{h_{t_{ki}}}; \overleftarrow{h_{t_{ki}}}]$. Features are concatenated to corresponding token's hidden representation i.e. $I_{t_{ki}} = [h_{t_{ki}} ; F_{ki}]$. Finally, $I_{t_{ki}}$ is passed through a hidden BiLSTM layer. The output of this BiLSTM layer is then fed to a softmax layer to obtain the predicted class for the token (similar as section 3.2). Pseudo code of our proposed approach is described in Algorithm 1.

4 | EXPERIMENTS

We performed a thorough analysis of the algorithms to analyze their performances. In this section, we provide the details of the experimental analysis.

[1]The term 'token' is more general term which includes word, mathematical symbols, punctuation symbols, etc

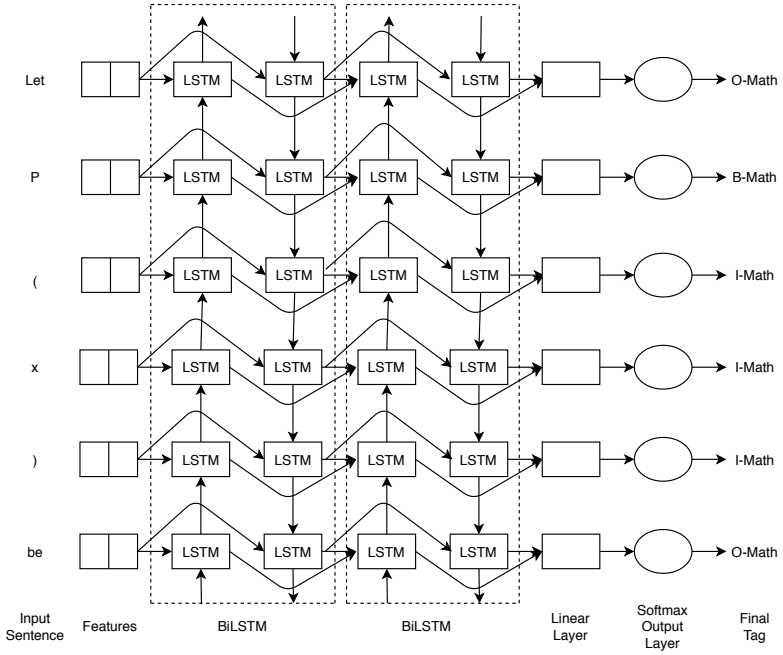

FIGURE 3 Architecture diagram for the proposed Bi-LSTM based Inline Mathematical Expression Detector (BIMExD)

Algorithm 1 Our proposed approach HIMExD

procedure HIMExD()
Input: Sequence of tokens S $=< t_1, t_2, \cdots, t_n >$
Output: a label to each of the t_i's (for $i \in \{1, \cdots, n\}$
  for `each sentence` in Scientific Document **do**
    for $t_i \in S$ **do**
      Extract a set of 17 features $F_i = (f_{i1}, f_{i2}, \ldots, f_{i17})$
      Pass t_i to an *embedding layer* to get its embedding representation E_{t_i}
      Fed E_{t_i} to Bi-LSTM layer to get $h_{t_i} = [\overrightarrow{h_{t_i}}; \overleftarrow{h_{t_i}}]$
      Concatenate features with token's hidden representation $I_{t_i} = [h_{t_i} ; F_i]$
      Pass I_{t_i} to BiLSTM layer
      Apply *softmax* to get the best tag for given t_i
    end for
  end for
end procedure

4.1 | Dataset

We adopted the dataset from (Iwatsuki et al. (2017)) for the experiments. The dataset is created from 70 PDF documents which are collected from ACL anthology[2]. Each PDF document or research paper has at least one mathematical expression, at most 699 mathematical expressions, and an average of 156 mathematical expressions. The average number of words in math-span is 3.61. Among all mathematical expressions, about 11.3% contain only one letter. More details about the annotation and the collection can be obtained from Iwatsuki et al. (2017). Few notable statistics are:

- Total Number of documents (papers): 70
- Total Number of tokens: 378,574
- Total number of tokens with label 'O-Math': 368,286
- Total number of tokens with label 'B-Math': 2,904

[2] `http://aclanthology.info`

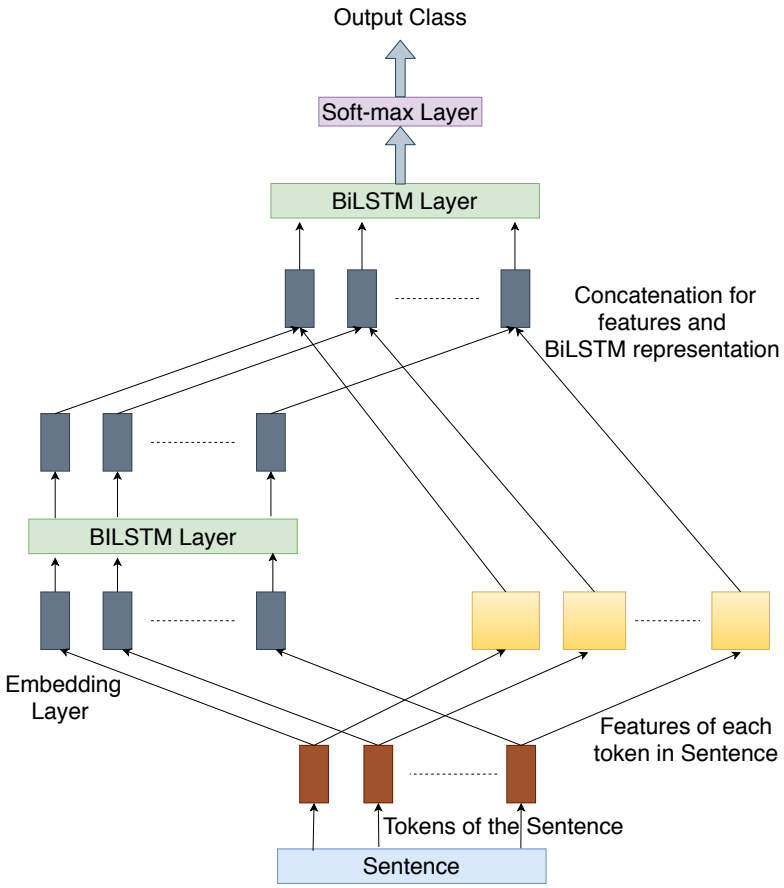

FIGURE 4 Architecture diagram for the proposed Hybrid neural network based Inline Mathematical Expression Detector (HIMExD)

- Total number of tokens with label 'I-Math': 7,378
- Maximum number of tokens in a single file: 8,616
- Minimum number of tokens in a single file: 2,466
- Average number of in-line expressions (math token per document): 146

4.2 | Evaluation Metrics

We used three evaluation metrics, namely, *precision*, *recall*, and *F-Measure*. Evaluation metric formulas and their descriptions are provided in Table 2. For all the three classes, we reported the macro and micro averages.

TABLE 2 Details of evaluation metrics (*tp* = true positives, *fp* = false positives and *fn* = false negatives)

Evaluation Metric	Formula	Description
Precision	$\text{Precision(P)} = \frac{tp}{tp+fp}$	What proportion of identified mathematical expressions are actually correct
Recall	$\text{Recall(R)} = \frac{tp}{tp+fn}$	What proportion of actual mathematical expressions was identified correctly
F-measure	$\text{F-Measure} = \frac{2.P.R}{P+R}$	Harmonic mean of precision and recall

4.3 | Baseline Models

The following baseline models are used to compare with our proposed approach.

- **InftyReader**[3] (Eto and Suzuki (2001)): This is a popular optical character reorganization(OCR)-based system to identify mathematical content in PDF documents. The system returns positioning attributes of the mathematical expressions for the given scientific document. The system is good for identification of independent math expressions but extremely bad for in-line expressions (we will provide sufficient evidence in subsequent sections).
- **CRF1** (Iwatsuki et al. (2017)): This baseline is close to our proposed feature-based model. It is based on feature engineering and CRF algorithm and was presented in Iwatsuki et al. (2017). However, we added some more extra features to improve the performance.
- **Bi-LSTM-CRF** (Huang et al. (2015)): This method uses CRF on top of Bi-LSTM model to get the tags for the given sequences.
- **Bi-LSTM-CRF+Features**: We developed this model as the combination of Bi-LSTM-CRF model and feature based method as described in 3.1.

TABLE 3 Results comparison of different models used in our experimentation. The first three models are from literature, whereas the remaining ones are proposed in this paper. CRF2 is our feature-based approach, BIMExD & HIMExD are our proposed deep learning based approaches. Bi-LSTM-CRF+Features is built on top of existing Bi-LSTM-CRF architecture.

(3a) Macro average

Method	Precision	Recall	F-Measure
InftyReader (Eto and Suzuki (2001))	13.8%	**86.7%**	23.8%
CRF1 (Iwatsuki et al. (2017))	86.5%	75.0%	80.4%
Bi-LSTM-CRF (Huang et al. (2015))	85.0%	70.1%	76.2%
Bi-LSTM-CRF+Features	88.5%	76.3%	81.6%
CRF2	87.8%	80.0%	83.4%
BIMExD	85.3%	82.6%	83.9%
HIMExD	**88.9%**	83.6%	**86.1%**

(3b) Micro average

Method	Precision	Recall	F-Measure
InftyReader (Eto and Suzuki (2001))	13.0%	89.9%	22.8%
CRF1 (Iwatsuki et al. (2017))	94.9%	83.6%	88.9%
Bi-LSTM-CRF (Huang et al. (2015))	98.4%	98.5%	98.5%
Bi-LSTM-CRF+Features	98.7%	98.6%	98.7%
CRF2	98.7%	**99.1**%	98.8%
BIMExD	**99.0%**	99.0%	**99.0%**
HIMExD	**99.0%**	**99.1%**	**99.0 %**

4.4 | Implementation Details

We used *sk-learn* CRF suite for implementation from official website[4] for feature based model i.e.,conditional random field (CRF). We applied single hidden layer bi-directional LSTM model in the all LSTM layers for both neural as well as hybrid model. For BIMExD model, we used learning rate= 0.1, embedding layer dimension = 50, LSTM hidden unit dimension= 200 (100 for each direction), number of epochs = 10 and batch size = 32. For HIMExD approach we used learning rate= 0.3, embedding layer dimension = 128, LSTM hidden unit dimension= 512 (256 for each direction), number of epochs = 10 (on single GPU machine) and dynamic batch size (i.e., batch size for each sentence is number of words in the sentence). All the trainable parameters including token embeddings are randomly initialized with $\mathcal{U}(-0.1, 0.1)$.

4.5 | Results and Discussion

The dataset consists of three different classes, i.e., B-Math: the beginning of the mathematical expression, I-Math: inside mathematical expression and O-Math: outside of mathematical expression. It can be observed from the dataset statistics that the dataset is highly imbalanced, and contains around 97% instances from the O-Math class. As a result, any method that performs well for the O-Math class, or blindly outputs the prediction as O-Maths class for all inputs will have very good overall accuracy. Due to this reason, apart from overall accuracy (micro-average), we also specifically analyze the performance of the algorithm for the minority classes: B-Math and I-Math.

As explained in the methodology section, three different proposed models are used for experiments CRF2, BIMExD, and HIMExD. We run 5-fold cross-validation for each model to provide fair results. Final evaluation scores of *precision, recall, and F-measure* for baseline models as well as our proposed models are shown in Table 3. Table 3a shows macro-average and Table 3b shows micro-average. *Micro-average* score takes data

[3]https://www.sciaccess.net/en/InftyReader/

[4]`https://sklearn-crfsuite.readthedocs.io/en/latest/#`

imbalance into the consideration and averages as per class proportion in the dataset. It takes a global token count (label proportion) into account. *Macro-average* score is a simple average score that does not consider the data imbalance factor. It is the average of the overall performances of the individual classes. Macro-averaging does not take into account a number of examples in each class, and hence can not be biased by a good performance on the majority class only. Scores of InftyReader indicate low confidence for predicting correct class label because of low precision and high recall. Other models are reported better scores for both precision and recall. CRF2 has significant improvement over CRF1 model which shows that the impact of including additional features (f-measure for CRF1 and CRF2 are: 80.4% & 83.4% for macro and 88.9% & 98.8% for micro average). CRF-2 performance is better than Bi-LSTM-CRF and Bi-LSTM-CRF+Features. Proposed neural model (BIMExD) and hybrid model (HIMExD) have achieved better evaluation scores than CRF2 model and other baseline methods (InftyReader, CRF1, Bi-LSTM-CRF, Bi-LSTM-CRF+Features), which shows the successful impact of deep learning models (as they capture a representation of important features efficiently). Bar charts comparing all the baseline and proposed methods for macro average and micro average are shown in Figure 5 and Figure 6 respectively. We can observe that our proposed methods F-Measure is performing better than all other baseline approaches.

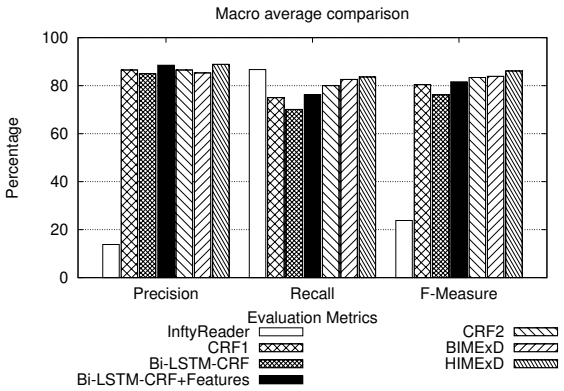

FIGURE 5 Macro average comparison

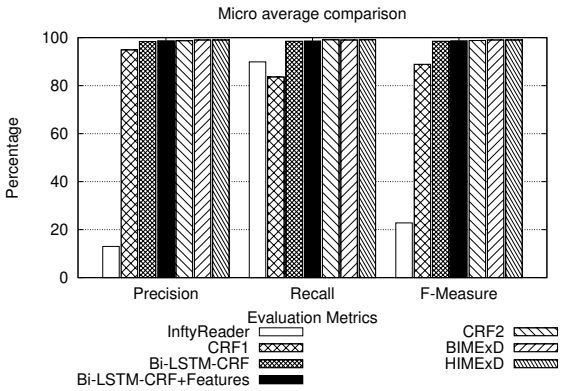

FIGURE 6 Micro average comparison

TABLE 4 Results of Conditional Random Fields -2 (CRF2) model

Class	Precision	Recall	F-Measure
B-Math	83.8%	65.2%	73.2%
I-Math	80.4%	75.0%	77.6%
O-Math	99.2%	99.8%	99.4%
Macro Average	87.8%	80.0%	83.4%
Micro Average	98.7%	99.1%	98.8%

TABLE 5 Results of BIMExD model

Class	Precision	Recall	F-Measure
B-Math	73.8%	67.8%	70.7%
I-Math	82.5%	80.4%	81.4%
O-Math	99.4%	99.6%	99.5%
Macro Average	85.3%	82.6%	83.9%
Micro Average	98.9%	98.8%	98.9 %

TABLE 6 Results of HIMExD model

Class	Precision	Recall	F-Measure
B-Math	79.5%	72.4%	75.8%
I-Math	87.9%	78.8%	83.1%
O-Math	99.4%	100%	100%
Macro Average	88.9%	83.6%	86.1%
Micro Average	99.0%	99.1%	99.0%

As we aim to achieve better evaluation results for minor classes (B-Math and I-Math), in this section, we will discuss the scores of each class in detail. The top three methods which are performed best according to the F-Measure (as described in Table 3) are analyzed further. Results of the Conditional Random Fields (CRF2) model are shown in Table 4. For all the evaluation metrics (*precision*, *recall*, *f-measure*) the score for O-Math class is around 99% which is expected because O-Math class has large number of training instances. For both B-Math and I-math classes *precision* is better than *recall*. *F-measure* is better for I-Math class compared to B-Math class. BIMExD results are shown in Table 5. The performance behavior for O-Math class in this method is similar to the behavior for O-Math class in the CRF method. *Precision* is better than *recall* for both B-Math and I-Math classes. In BIMExD, *F-Measure* for I-Math class is higher compared to B-Math class.

Overall results and observation of hybrid model HIMExD (presented in Table 6) are similar to the BIMExD model. Possible reasoning behind this is that the model retains the benefits of the carefully selected features (as done in CRF-2) and the automated feature engineering (as done in BIMExD), and the combination leads to a better performance. The feature based model CRF-2 had a poor recall for the B-Math class, which indicates that the features were not able to detect many actual B-Math instances. Since the precision of O-Math is almost 100%, some of these false negatives for B-Math are actually being detected as I-Math - which may still be Ok as the complete chain of the mathematical expressions will still be detected. Performance for O-Math class is boosted even further, and all examples for this class are successfully detected by the method, resulting in a 100% recall. CRF2 precision is higher than the precision of HIMExD for B-Math class and BIMExD recall is better than recall of HIMExD for I-Math class. However, HIMExD is better than CRF2 and BIMExD models in terms of F-measure for all the classes. Overall, the proposed neural methods (BIMExD and HIMExD) achieve significantly better performances over the baseline approaches taken from literature.

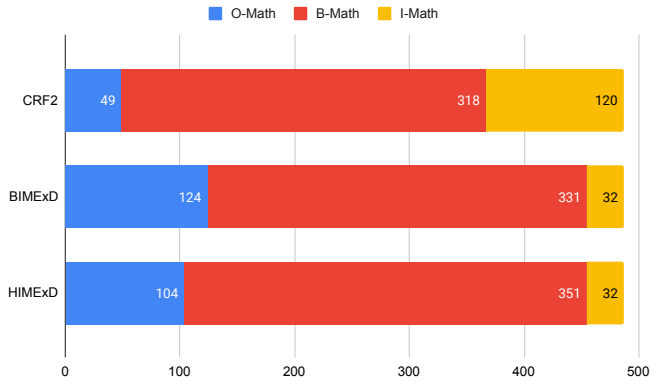

FIGURE 7 B-Math confusion matrix comparison

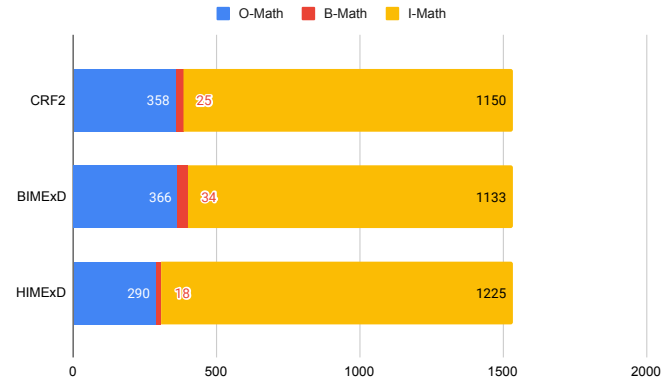

FIGURE 8 I-Math confusion matrix comparison

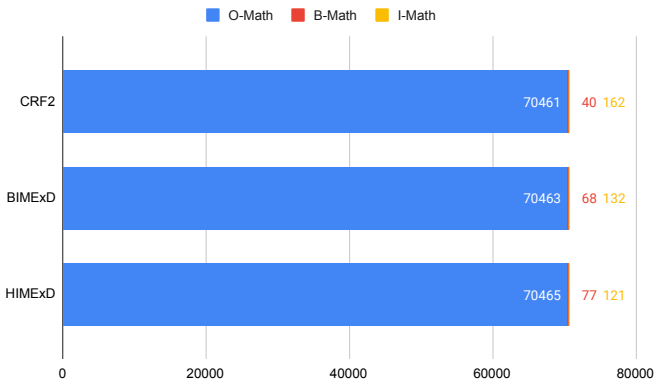

FIGURE 9 O-Math confusion matrix comparison

The top three best performing methods according to F-Measure (described in Table 3) are selected for confusion matrix comparison. The confusion matrix comparison for three methods CRF2, BIMExD, and HIMExD for B-Math class is shown in Figure 7. We can observe that a higher number of B-Math instances are predicted as O-Math instances in BIMExD compared to CRF2 and HIMExD. Our proposed approach HIMExD predicted a higher number of B-Math instances correctly compared to CRF2 and BIMExD. For BIMExD and HIMExD methods the number of B-Math instances predicted as I-Math are lower in number whereas for CRF2 large number (120) of such instances are predicted as I-Math. So, in CRF2 the proposed features are in confusion between B-Math and I-Math. Confusion matrix comparison for I-Math class is shown in Figure 8. Similar behavior is observed for I-Math class as B-Math class i.e., more I-Math instances are predicted as O-Math in BIMExD compared to CRF2 and HIMExD. Similarly, more I-Math instances are predicted as I-Math for HIMExD compared to CRF2. Confusion matrix comparison for O-Math class is shown in Figure 9. All three methods are predicting O-Math instances O-Math most of the times. This is due to more number of O-Math instances are available for training.

In summary, our proposed approach outperforms all the methods in the literature which indicates that the method can be used to successfully detect inline mathematical expressions from scientific documents.

4.5.1 | Result analysis: Different cases

We now perform an in-depth analysis of the failure cases of the algorithms. We see that all the methods except InftyReader perform very well for the O-Math class. The F-Scores for the O-Math class are above 99% for most of the methods indicating both the precision and recall for the O-Math class are high. On the other hand, the performances of the methods vary for the classes that have mathematical symbols. Hence, in this analysis, we look closely at the test cases containing the mathematical symbols. We characterized and categorized the math symbols or expressions in multiple groups, as shown in the Table 7. In the table, we also indicate the performances of the methods for expressions of these types. If a method misclassifies examples from a particular type for more than 70% of the test cases, we consider that the method performs poorly (or fail) for that type. Otherwise, the method is considered to do well (or succeed in identifying) examples from that type.

We can observe that our methods BIMExD and HIMExD can detect most of the mathematical expressions compared to literature method BiLSTM-CRF, baseline method BiLSTM-CRF + Features and other feature based method CRF-2. If mathematical symbol (such as _, -, =, etc.) is present between two variables or between variable and number or if there are any assignment expressions such as $n = 1$, $m = 2$, $k = 3$, etc. then our proposed method HIMExD performs better than BIMExD, BiLSTM-CRF, CRF-2, and BiLSTM-CRF + Features. This indicates that our proposed features combined with the BIMExD model are good indicators to detect these kinds of expressions. The deep learning method BIMExD alone is not sufficient to handle these types of expressions. If the expression type is long probability expression like $P(e_5|e_4, e_3, C_{a5} = f_6, f_7, f_8), P(e_5|e_4, e_3, C_{a4} = f_8, f_9, f_{10}), P(o(f_7) = \langle o_{L7}(f_7) = RA, o_{R7}(f_7) = RA\rangle)$, etc. or if the expression is long expression like $I(f_j) = e_{b_j} - d, ..., e_{b_j}, ..., e_{b_j} + d$ then these type of expressions are misclassified by BiLSTM-CRF, CRF-2, and BiLSTM-CRF + Features whereas our proposed methods BIMExD and HIMExD classified them correctly. Moreover, the baseline and literature methods fail to detect the double subscripts and special symbols for more than 50% of the test cases. On the other hand, for whatever expression types the proposed methods are successful (marked with a ✔ in Table 7), at least 85% of the examples from those individual types are correctly detected.

For the following types of test cases, all methods failed to detect them correctly.

- If the expression contains long variable names such as $bestScore(n + 3, \langle end_1, end_2, end_3\rangle)$, $cache(i + 2, \langle s_{i-1}, s_i, s_{i+1}\rangle)$, etc.
- If there are word variables like $Vocab_i$, $Corpus_i$, $p - value$, M1, M2 i.e. long variable names having more than one character in length and
- If the *log* expressions present in between | operator

The following are some more failure cases of CRF-2 apart from the cases described above.

- If a variable list has ellipsis in between (e.g. $x_1 \cdots x_k$, $y_1 \cdots y_k$), then it is misclassified as O-Math.
- If there is comma inside variable or symbol list (e.g. expressions in the form $< e, c >$ and $< a_1, a_2 >$), then it is misclassified as O-Math. As a result the variable after comma is also misclassified as B-Math because the system considers the expression after comma as a new expression.
- Variables like a, b, c in list numbering $(a), (b), (c)$ are misclassified as B-Math.

From the above observations we can say that our proposed method detects most of the inline mathematical expressions compared to baseline and literature methods.

TABLE 7 Error Analysis for different methods. If an entry (Expression_Type_i, Method_j) contains "✕" symbol then it indicates that Method_j failed to detect more than 70% of test cases from that expression type. If Method_j is able to correctly detect expressions from Expression_Type_i for at least 70% of the cases, we consider the method to have succeeded for that expression category and mark the corresponding cell with a ✔.

Expression Type	**BiLSTM-CRF**	**CRF-2**	**BiLSTM-CRF + Features**	**BIMExD**	**HIMExD**
Mathematical symbol between two variables or one variable and number i.e. i-k, j-k, 1-σ, t+1, P+Q, etc.	×	×	×	×	✔
Assignment expressions such as m=3, n=1, k=4, etc.	×	×	×	×	✔
Long Probability expressions such as $P(e_5\|e_4, e_3, C_{a5} = f_6, f_7, f_8)$, $P(e_5\|e_4, e_3, C_{a4} = f_8, f_9, f_{10})$, $P(o(f_7) = \langle o_{L7}(f_7) = RA, o_{R7}(f_7) = RA\rangle)$	×	×	×	✔	✔
Long expressions: $I(f_j) = e_{b_j} - d, ..., e_{b_j}, ..., e_{b_j} + d$	×	×	×	✔	✔
Expressions which contain long variable names such as $\text{bestScore}(n + 3, \langle end_1, end_2, end_3\rangle)$; $\text{cache}(i + 2, \langle s_{i-1}, s_i, s_{i+1}\rangle)$	×	×	×	×	×
Word variables like Vocab_i, Corpus_i, $p - value$, etc.	×	×	×	×	×
Two-letter variables like M1, M2	×	×	×	×	×
Expressions like |logZ|, |logk|	×	×	×	×	×

5 | CONCLUSION

In this paper, we propose a hybrid approach that makes use of both feature based method and neural network based method. Due to large lexical similarity between the tokens of in-line mathematical expressions and tokens of the regular text, different articles (a, an, A, etc..), pronouns (I), etc. the problem of detecting inline mathematical expressions from scientific documents is challenging. We formulated this problem as a sequence tagging problem and used BIO (Beginning-Inside-Outside) encoding for labeling. Further, we used feature-based model, BIMExD and HIMExD approaches. Dataset is highly unbalanced, so our focus was to improve evaluation scores of the minority classes. Our results show that proposed models outperformed baselines with a large margin. Still there are few expression types for which the proposed methods fail to detect good number of the test cases. We will look to improve the prediction accuracy for those expression types. For future work, it will be interesting to explore the pre-training of token embeddings, which will give contextual representation of tokens and is expected to boost the evaluation scores even further.

ACKNOWLEDGEMENT

The first author of this paper is supported by a fellowship from Visvesvaraya PhD Scheme of Ministry of Electronics and Information Technology, Government of India with grant number EE/2015-16/023/MLB/MZAK/0176. Part of the work was done as part of an internship at The University of Tokyo under Sakura Science plan.

References

Akbik, A., Blythe, D., & Vollgraf, R. (2018). Contextual string embeddings for sequence labeling. In *Proceedings of the 27th international conference on computational linguistics* (pp. 1638–1649).

Alkalai, M., Baker, J. B., Sorge, V., & Lin, X. (2013). Improving formula analysis with line and mathematics identification. In *2013 12th international conference on document analysis and recognition* (pp. 334–338).

Alzaidy, R., Caragea, C., & Giles, C. L. (2019). Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents. In *The world wide web conference* (pp. 2551–2557).

Chan, C. (2019). Stroke extraction for offline handwritten mathematical expression recognition. *arXiv preprint arXiv:1905.06749*.

Chu, W.-T., & Liu, F. (2013). Mathematical formula detection in heterogeneous document images. In *2013 conference on technologies and applications of artificial intelligence* (pp. 140–145).

Constantin, A., Pettifer, S., & Voronkov, A. (2013). Pdfx: fully-automated pdf-to-xml conversion of scientific literature. In *Proceedings of the 2013 acm symposium on document engineering* (pp. 177–180).

Drake, D. M., & Baird, H. S. (2005). Distinguishing mathematics notation from english text using computational geometry. In *Eighth international conference on document analysis and recognition (icdar'05)* (pp. 1270–1274).

Eto, Y., & Suzuki, M. (2001). Mathematical formula recognition using virtual link network. In *Proceedings of sixth international conference on document analysis and recognition* (pp. 762–767).

Futrelle, R. P., Shao, M., Cieslik, C., & Grimes, A. E. (2003). Extraction, layout analysis and classification of diagrams in pdf documents. In *Seventh international conference on document analysis and recognition, 2003. proceedings.* (pp. 1007–1013).

Gao, J., Galley, M., & Li, L. (2019). *Neural approaches to conversational ai: Question answering, task-oriented dialogues and social chatbots*. Now Foundations and Trends.

Garain, U. (2009). Identification of mathematical expressions in document images. In *2009 10th international conference on document analysis and recognition* (pp. 1340–1344).

Garain, U., & Chaudhuri, B. (2000). A syntactic approach for processing mathematical expressions in printed documents. In *Proceedings 15th international conference on pattern recognition. icpr-2000* (Vol. 4, pp. 523–526).

Garain, U., Chaudhuri, B., & Chaudhuri, A. R. (2004). Identification of embedded mathematical expressions in scanned documents. In *Proceedings of the 17th international conference on pattern recognition, 2004. icpr 2004.* (Vol. 1, pp. 384–387).

Huang, Z., Xu, W., & Yu, K. (2015). Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Iwatsuki, K., Sagara, T., Hara, T., & Aizawa, A. (2017). Detecting in-line mathematical expressions in scientific documents. In *Proceedings of the 2017 acm symposium on document engineering* (pp. 141–144).

Jin, J., Han, X., & Wang, Q. (2003). Mathematical formulas extraction. In *Icdar* (pp. 1138–1141).

Kacem, A., Belaïd, A., & Ahmed, M. B. (2001). Automatic extraction of printed mathematical formulas using fuzzy logic and propagation of context. *International Journal on Document Analysis and Recognition*, *4*(2), 97–108.

Lin, X., Gao, L., Tang, Z., Hu, X., & Lin, X. (2012). Identification of embedded mathematical formulas in pdf documents using svm. In *Document recognition and retrieval xix* (Vol. 8297, p. 82970D).

Lin, X., Gao, L., Tang, Z., Lin, X., & Hu, X. (2011). Mathematical formula identification in pdf documents. In *2011 international conference on document analysis and recognition* (pp. 1419–1423).

Lipinski, M., Yao, K., Breitinger, C., Beel, J., & Gipp, B. (2013). Evaluation of header metadata extraction approaches and tools for scientific pdf documents. In *Proceedings of the 13th acm/ieee-cs joint conference on digital libraries* (pp. 385–386).

Liu, N., Zhang, D., Xu, X., Guo, L., Chen, L., Liu, W., & Ke, D. (2017). Robust math formula recognition in degraded chinese document images. In *2017 14th iapr international conference on document analysis and recognition (icdar)* (Vol. 1, pp. 113–118).

Lopez, P. (2009). Grobid: Combining automatic bibliographic data recognition and term extraction for scholarship publications. In *International conference on theory and practice of digital libraries* (pp. 473–474).

Ma, X., & Hovy, E. (2016). End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

Mansouri, B., Rohatgi, S., Oard, D. W., Wu, J., Giles, C. L., & Zanibbi, R. (2019). Tangent-cft: An embedding model for mathematical formulas. In *Proceedings of the 2019 acm sigir international conference on theory of information retrieval* (pp. 11–18).

Ohyama, W., Suzuki, M., & Uchida, S. (2019). Detecting mathematical expressions in scientific document images using a u-net trained on a diverse dataset. *IEEE Access*, *7*, 144030–144042.

Peng, N., & Dredze, M. (2016). Multi-task domain adaptation for sequence tagging. *arXiv preprint arXiv:1608.02689*.

Phong, B. H., Hoang, T. M., & Le, T.-L. (2019). Mathematical variable detection based on convolutional neural network and support vector machine. In *2019 international conference on multimedia analysis and pattern recognition (mapr)* (pp. 1–5).

Phong, B. H., Hoang, T. M., Le, T.-L., & Aizawa, A. (2019). Mathematical variable detection in pdf scientific documents. In *Asian conference on intelligent information and database systems* (pp. 694–706).

Ramírez-Piña, C., Sánchez, J. S., Valdovinos-Rosas, R. M., & Hernández-Servín, J. A. (2018). A hybrid feature extraction method for offline handwritten math symbol recognition. In *Iberoamerican congress on pattern recognition* (pp. 893–901).

Rei, M. (2017). Semi-supervised multitask learning for sequence labeling. *arXiv preprint arXiv:1704.07156*.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention* (pp. 234–241).

Saggion, H., & Ronzano, F. (2016). Natural language processing for intelligent access to scientific information. In *Proceedings of coling 2016, the 26th international conference on computational linguistics: Tutorial abstracts* (pp. 9–13).

Seo, M., Kembhavi, A., Farhadi, A., & Hajishirzi, H. (2016). Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Simpson, E. D., & Gurevych, I. (2019). A bayesian approach for sequence tagging with crowds. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp)* (pp. 1093–1104).

Singh, M., Barua, B., Palod, P., Garg, M., Satapathy, S., Bushi, S., … others (2016). Ocr++: a robust framework for information extraction from scholarly articles. *arXiv preprint arXiv:1609.06423*.

Tian, X., Yu, B., & Sun, J. (2014). A preprocessing and analyzing method of images in pdf documents for mathematical expression retrieval. *TELKOMNIKA Indonesian Journal of Electrical Engineering*, *12*(6), 4579–4588.

Tkaczyk, D., Szostek, P., Fedoryszak, M., Dendek, P. J., & Bolikowski, Ł. (2015). Cermine: automatic extraction of structured metadata from scientific literature. *International Journal on Document Analysis and Recognition (IJDAR)*, *18*(4), 317–335.

Tomanek, K., & Hahn, U. (2009). Semi-supervised active learning for sequence labeling. In *Proceedings of the joint conference of the 47th annual meeting of the acl and the 4th international joint conference on natural language processing of the afnlp: Volume 2 - volume 2* (p. 1039–1047). USA: Association for Computational Linguistics.

Tran, G. S., Huynh, C.-K., Le, T.-S., Phan, T.-P., & Bui, K.-N. (2018). Handwritten mathematical expression recognition using convolutional neural network. In *2018 3rd international conference on control, robotics and cybernetics (crc)* (pp. 15–19).

Xu, C., Tang, Z., Tao, X., Li, Y., & Shi, C. (2013). Graph-based layout analysis for pdf documents. In *Imaging and printing in a web 2.0 world iv* (Vol. 8664, p. 866407).

Yang, Z., Salakhutdinov, R., & Cohen, W. W. (2017). Transfer learning for sequence tagging with hierarchical recurrent networks. *arXiv preprint arXiv:1703.06345*.

Zanibbi, R., Aizawa, A., Kohlhase, M., Ounis, I., Topic, G., & Davila, K. (2016). Ntcir-12 mathir task overview. In *Ntcir*.

Zhang, J., Du, J., & Dai, L. (2018a). Multi-scale attention with dense encoder for handwritten mathematical expression recognition. In *2018 24th international conference on pattern recognition (icpr)* (pp. 2245–2250).

Zhang, J., Du, J., & Dai, L. (2018b). Track, attend, and parse (tap): An end-to-end framework for online handwritten mathematical expression recognition. *IEEE Transactions on Multimedia*, *21*(1), 221–233.

Zhang, T., Mouchère, H., & Viard-Gaudin, C. (2018). A tree-blstm-based recognition system for online handwritten mathematical expressions. *Neural Computing and Applications*, 1–20.