1K2-IOS-1b-6

Automatic Approach to Understanding Mathematical Expressions Using MathML Parallel Markup Corpora

Minh-Quoc Nghiem[*1] Giovanni Yoko[*2] Yuichiroh Matsubayashi[*3] Akiko Aizawa[*2*3]

[*1] The Graduate University for Advanced Studies [*2] The University of Tokyo
[*3] National Institute of Informatics

This paper explores the use of MathML Parallel Markup Corpora for automatic understanding of mathematical expressions, the task of which is formulated as a translation from Presentation to Content MathML Markups. In contrast to previous research that mainly relied on manually encoded transformation rules, we use a statistical-machine-translation-based method to automatically extract translation rules from parallel markup corpora. Our study shows that the structural features embedded in the MathML tree can be effectively exploited in the subtree alignment and the translation rules extracted from the alignment give a boost to the translation system. Experimental results on the Wolfram Function Site show that our approach is an improvement over a prior rule-based system.

1. Introduction

1.1 Motivation

One of the most significant discussions regarding the digitization of mathematical and scientific content and its applications is about semantic enrichment of mathematical documents, that is, adding or associating semantic tags - usually concepts - with mathematical expressions. By encoding the underlying mathematical meaning of an expression explicitly, it is possible to interchange information more precisely between systems that semantically process mathematical objects. The direct application of this idea enables semantic searches for mathematical expressions whereby the systems understanding of the intent of the searcher and the contextual meaning of mathematical terms improves search accuracy. It also benefits computer algebra systems, automatic reasoning systems and multi-lingual translation systems.

However, as is the case with natural language, semantic enrichment of mathematical expressions is a non-trivial task.

- First, mathematical notation, though more rigorous than natural language, is nonetheless at times ambiguous, context-dependent, and varies from community to community. [4] points out that the difficulties in inferring semantics from a presentation stem from the fact that there are many-to-one mappings from a presentation to semantics and vice versa.
- Second, the underlying mathematical meaning of an expression needs to follow a semantic markup in a semantically rigorous way. Because of this, in failing to follow the constraint, the computer might not be able to process that expression.
- The third problem is that new notations tend to be introduced and used when needed so a mechanism is required for referring to mathematical concepts outside of the base collection.

The aim of this paper is to introduce a method of automatic semantic enrichment for mathematics that is capable of analyzing and disambiguating mathematical terms.

1.2 Problem Statement

In our research, MathML [4] Presentation Markup is used to display mathematical expressions and MathML Content Markup is used to convey mathematical meaning. The semantic enrichment task then becomes one of generating Content MathML outputs from Presentation MathML expressions.

There are three reasons why we chose MathML markup in our research.

- First, since its first release in 1997, MathML has grown to become a general format that enables mathematics to be served, received, and processed in a wide variety of applications.
- Second, MathML can be used to encode both mathematical notations and mathematical content.
- Last, large collections of formulas are available in MathML, and we can easily assess these collections.

In the scope of this paper, we only make use the information within a mathematical expression for disambiguation when translating it into content markup.

1.3 Limitations of Prior Work

The prior solution to this problem is SnuggleTeX [5], which was proposed by David McKain. The system uses rule-based methods for disambiguation and translation. This solution has two main limitations:

- Since it is a hand-written rule-based system, SnuggleTeX requires mathematical knowledge and human effort to develop.
- Due to the diversity of mathematical expressions, SnuggleTeX is still considered experimental and has difficulty processing complicated mathematical symbols and expressions.

1.4 Our Approach and Key Contributions

In this paper, we propose an approach that automatically learns semantic inferences in a presentation from parallel markup data. This approach is based on statistical machine translation. The underlying mathematical meaning of an expression is inferred from the probability distribution $p(c|p)$ that a semantic expression c is the translation of a presentation expression p. The probability distribution is automatically learned from both Presentation and Content MathML markup data, that is, parallel markup MathML data. The data used in this study was collected from the Wolfram Function Site [14]. We also prepared other parallel markup MathML data by annotating mathematical expressions in 20 papers from The Archives of the Association for Computational Linguistics [15] (ACL-ARC).

There are two main contributions in this paper:

- First, we successfully applied machine translation techniques to solving the problem of mathematic semantic enrichment. Experimental results show that our system significantly outperforms the current rule-based system and it can handle a lot of practical cases in the semantic enrichment problem. The quantity and quality of mathematical expressions are continuing to grow, and we believe that our system will be able to cover most mathematical expressions.

- Second, mathematics knowledge such as a symbol's meanings or structural relations is automatically learned while training; therefore, the system requires no human effort or expertise, and it is easier to update with more data. Since new notations keep cropping up, it is important to update the system as quickly as possible.

1.5 Summary of Experimental Results

We performed a ten-fold cross validation on mathematical expressions from six categories of the Wolfram Functions Site to evaluate the effectiveness of our learning method. We performed another experiment to assess the correlation between the systems performance and training set size and found that increasing the size of the training data boosted the systems performance. We also performed an extensive comparison with prior work [5] using a data set collected from ACL-ARC scientific papers. Our experimental results show that our approach works well in dealing with the mathematics semantic enrichment problem and it outperforms the previous work by making significantly fewer errors.

The remainder of this paper is organized as follows: In Section 2, we give a brief overview of the background and related work on semantic enrichment of mathematical expressions. We present our method in Section 3 and describe the experimental setup and results in Section 4. Section 5 concludes the paper and gives avenues for future work.

2. Related Work

2.1 MathML on the Web

Since mathematical formulas contain both mathematical symbols and structures, a special markup is required for their representation. Until recently, images have been used to represent mathematical formulas on the web. This type of display does not need any markup language to decode the formulas, but it is hard to process them. A way of dealing with mathematical formulas in this format is to convert them into another text-based format, for example, InftyReader [2].

TEX has been used to encode mathematical formulas in scientific documents. TEX is popular in academia, especially mathematics, since it provides a text syntax for mathematical formulas. A formula is printed in a way a person would write by hand, or typeset the equation. In some web pages, such as on the Wikipedia site, formulas are displayed in both image and TEX formats.

The best known open markup format for representing mathematical formulas for the web is MathML [4], which was recommended by the W3C math working group. It provides a standard way of representing mathematical expressions. It is an XML application for describing mathematical notations and encoding mathematical content within a text format. MathML has two types of encoding, content-based encoding, called Content MathML, dealing with the meaning of formulas, and presentation-based encoding, called Presentation MathML, dealing with the display of formulas. The illustration trees of the Presentation and Content Markup of the expression $arctan(0) = 0$ are depicted in Figure 1 and Figure 2. Besides MathML, there are other markups such as eqn [13], OpenOffice.org Math [10], ASCIIMathML [11], and OpenMath [12], but these markups can be converted into MathML by using freely available tools.

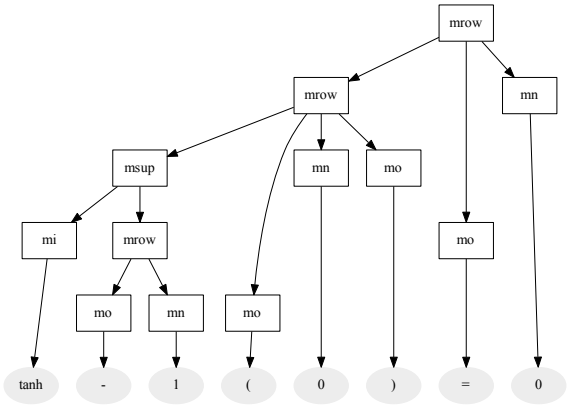

Figure 1: Tree display of Presentation Markup of the expression $arctan(0) = 0$

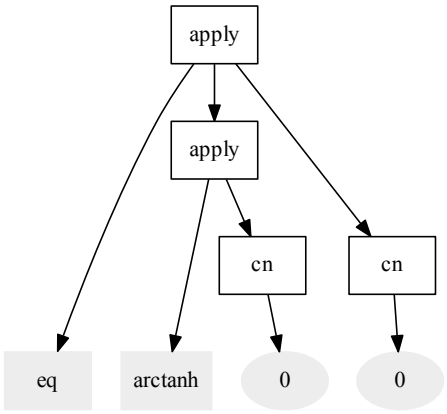

Figure 2: Tree display of Content Markup of the expression $arctan(0) = 0$

2.2 Systems with Semantic Enrichment Feature

There are not many studies on the semantic enrichment problem. In this section, we list some of the work on exploiting the meanings of mathematical expressions.

Grigole et al. [3] proposed an approach to understanding mathematical expressions based on the text surrounding the mathematical expressions. The main idea of this approach is to use the surrounding text for disambiguation based on word sense disambiguation and lexical similarity. First, a local context C (five nouns preceding a target mathematical expression) is found in each sentence. For each noun, the system identifies a Term Cluster (derived from the OpenMath Content Dictionary) with the highest semantic similarity according to a similarity metric. The similarity scores obtained are weighted, summed up, and normalized by the length of the considered context. The Term Cluster with the highest similarity score is assigned as the interpretation. The approach was evaluated on 451 manually annotated mathematical expressions, and the best result was an $F_{0.5}$ score of 68.26. To deal with the meanings of mathematical formulas, Nghiem et al. [7] proposed an approach for extracting names or descriptions of formulas by using the natural language text surrounding them. The most accurate extraction result using data from Wikipedia was 68.33 percent.

There are two other projects that deal with the semantic meaning of mathematical expressions. The first is the SnuggleTeX project [5], which provides a free and open-source Java library for converting fragments of LaTeX into XML including Content MathML. The other project is Lamapun [6]. This project investigates semantic enrichment, structural semantics, and ambiguity resolution in mathematical corpora. Unfortunately, there are no evaluations of these systems.

2.3 Method for automatically extracting translation rules from data

To translate mathematical expressions from the Presentation MathML into Content MathML format, a list of translation rules is required. Building these translation rules by hand is a large undertaking. Our task is inherently domain-specific; therefore, we devised an approach based on statistical machine learning for automatically extracting rules from a dataset.

Statistical machine translation (SMT) is by far the most widely studied machine translation method. SMT uses a very large data set of good translations, that is, a corpus of texts which have already been translated into another language, and it uses those texts to automatically infer a statistical model of translation. The statistical model is then applied to new texts to make a translation of them. Tree-based or syntax-based SMT can be used for tree-to-tree translation but it has two drawbacks when it is applied to the problem of translating Presentation into Content MathML.

- The first drawback is that tree-based SMT focuses on generating surface texts rather than tree structures. Mathematical expressions have strict structures, and it fails to fulfill this requirement.
- The second drawback is there are many long mathematical expressions in real-world data and translating long and complex sentences has been a critical problem in machine translation.

To overcome these limitations, we made two separate rule sets: fragment rules and translation rules. The details are described in the next section.

3. Methodology

3.1 System Overview

The framework of the system is shown in Figure 3. The system has three main modules.

- Preprocessing: This module processes MathML expressions by removing error expressions or format tags with no semantic meaning.
- Rule Extraction: This module is given a dataset containing MathML parallel markup expressions, and it extracts translation rules from the dataset.
- Content MathML Generation: This module is given mathematical expressions in Presentation MathML markup and a set of rules, and it generates Content MathML expressions to enrich the Presentation MathML expressions.

3.2 Preprocessing

The presentation elements of Presentation MathML are divided into two classes: token elements and layout schemata. Token elements represent the identifier's names,

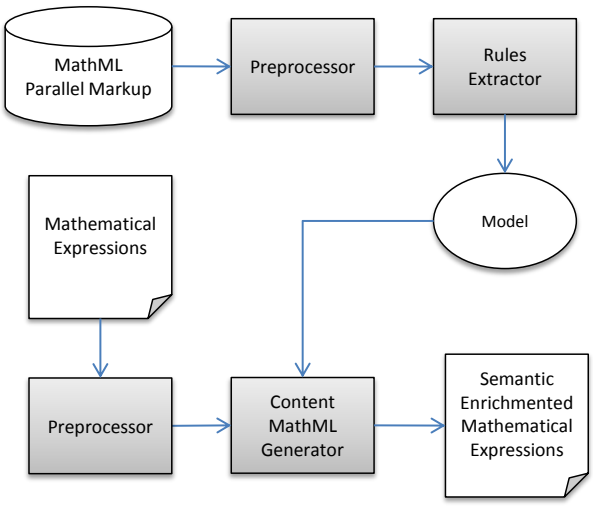

Figure 3: System Framework

function's names, numbers, etc. Layout schemata build expressions out of parts. After investigating data on the Wolfram Function Site, we noticed that there are elements that have no specific meaning; they are used for display purposes only and most of them are layout schemata. For example, the $< mtext >< /mtext >$ or $< mspace/ >$ tags are used to insert some space between expressions. Another example is pairs of parentheses; these are used to indicate that the expressions in the parentheses go together, despite that their structure already encodes that information. This preprocessing step removes these elements. We also remove mathematical expressions with error markups such as expressions that have no Content markup. For simplification, expressions with more than 200 content nodes are also removed.

3.3 Extracting Rules

In the training phase, we use GIZA++ [1] for aligning Presentation MathML terms and Content MathML terms. Based on the aligned data, we use heuristics to extract rules that we call "fragment rules". Fragment rules are rules that define the translation from the Presentation MathML sub-trees to the Content MathML sub-trees. These rules are used to break up a large Presentation MathML tree into smaller sub-trees while maintaining the structure of the output Content MathML trees. These rules are extracted based on the fact that translating a small tree is easier than translating a large one. Each rule in the fragment rule set is associated with a probability, that is, the frequency at which a rule occurs in the training data. For the expression $arctan(0) = 0$, the fragment rule is $mrow\{mrow[1]mo(=)[0]mn[2]\} \rightarrow apply\{eq[0]apply[1]cn[2]\}$. The numbers indicate which subtrees should go together between the two trees. The rule is depicted in Figure 4.

Once the sub-trees cannot be broken down further, we start to extract other rules, which we call "translation rules". We enhance the translation rule set with translation terms extracted by GIZA++. The pseudo code for

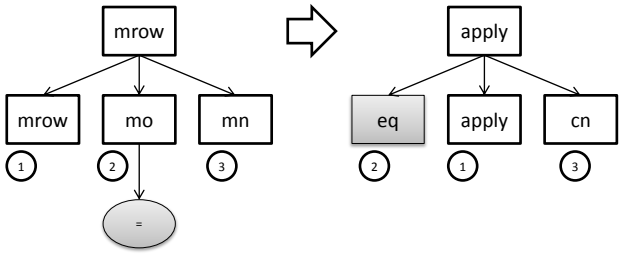

Figure 4: An example of fragment rule

extracting fragment rules is described in Algorithm 1.

Algorithm 1 Extract Fragment Rule

Input: a set of training MathML files parallel markup M
Output: a list of fragment rules FR
Output: a list of translation rules TR

```
FR ← ∅
TR ← ∅
A ← Alignment(M)
repeat
  for all m ∈ M do
    fr, tr ← ExtractRule(m, A)
    FR ← FR ∪ {fr}
    TR ← FR ∪ {tr}
  end for
  M ← ApplyRule(FR, M)
until NewRule(FR) = 0
return FR, TR
```

Table 1 shows examples of translation rules, and Table 2 shows examples of fragment rules.

Table 1: Examples of translation rules

< mo > = < /mo > → < eq/ >
< mo > . < /mo > → < times/ >
< mo > /; < /mo > → < ci > Condition < /ci >
< mo > ∈ < /mo > → < in/ >
< mi > n < /mi > → < ci > n < /ci >
< mn > 0 < /mn > → < cn type="integer" > 0 < /cn >
< mo > == < /mo > → < eq/ >
< mi > m < /mi > → < ci > m < /ci >
< mn > 1 < /mn > → < cn type="integer" > 1 < /cn >
< mo > - < /mo > → < plus/ >

3.4 Generating Content MathML

In the previous steps, we get two sets of rules, a fragment rule set and a translation rule set. We then use these rules for translation. Given mathematical expressions in Presentation MathML markup, the system will generate Content MathML markup for each expression.

- First, the expression is preprocessed to remove non-semantic elements.
- Second, the fragment rule is applied to the expression until it cannot be divided any further.

Table 2: Examples of fragment rules

mrow { mrow[1] mo(=)[0] mrow[2] } → apply { eq[0] apply[1] apply[2] }
mrow { mrow[1] mo(/;)[0] mrow[2] } → apply { ci(Condition)[0] apply[1] apply[2] }
mrow { msup[1] mo(.)[0] mrow[2] } → apply { times[0] apply[1] apply[2] }
mrow { mrow[1] mo(==)[0] mrow[2] } → apply { eq[0] apply[1] list[2] }
mrow { mrow[1] mo($\propto$)[0] mrow[2] } → apply { ci(Proportional)[0] apply[1] apply[2] }

- Third, the small sub-expressions in Presentation MathML markup are translated into sub-expressions in Content MathML markup by using the translation rule set. If no translation rule is found for a sub-expression, that expression is marked as untranslated.
- Last, sub-expressions in Content MathML markup are grouped to form the complete Content MathML expression.

Before the last step, we add a heuristic translation to translate numbers and identifiers in the *mn* and *mi* tags. The translation algorithm is described in Algorithm 2.

Algorithm 2 Translate Presentation to Content MathML tree

Input: a Presentation MathML tree tp
  fragment rules FR
  translation rules TR
Output: a Content MathML tree tc
  $L \leftarrow \emptyset$
  $TP \leftarrow \{\text{tp}\}$
  while $TP \neq \emptyset$ **do**
    for all $t \in TP$ **do**
      $TP \leftarrow TP \setminus \{t\}$
      if $CanNotApplyRule(t, FR)$ **then**
        $L \leftarrow L \cup \{t\}$
      else
        $TP = TP \cup ApplyRule(t, FR)$
      end if
    end for
  end while
  $L' \leftarrow \emptyset$
  for all $l \in L$ **do**
    $L' \leftarrow L' \cup Translate(l, TR)$
  end for
  tc $\leftarrow RebuildTree(L', FR)$
  return tc

4. Experimental Results

4.1 Evaluation Setup

The experiments were carried out using datasets from the Wolfram Function site. This site was created as a resource for educational, mathematical, and scientific communities. It contains the world's most encyclopedic collection of information about mathematical functions. All formulas on this site are available in both Presentation MathML and Content MathML format. The datasets we used contain 205,653 mathematical expressions belonging to six categories. All of these expressions have both MathML Presentation and Content Markups.

Training and testing were performed using ten-fold cross-validation; for each category, the original corpus was partitioned into ten subsets. Of the ten subsets, a single subset was retained as the validation data for testing the model, and the remaining subsets were used as training data. The cross-validation process was repeated ten times, with each of the ten subsets used exactly once as the validation data. The ten results from the folds then were averaged to produce a single estimation.

To prove the effectiveness of our models with real data, we conducted another experiment on the mathematical expressions in scientific papers. Currently, we have 20 papers from the ACL archive, and we manually annotated all of the math expressions in these papers with both Presentation Markup and Content Markup. We called this data ACL-ARC. In the first experiment, the data was not compatible with SnuggleTeX since SnuggleTeX uses ASCII MathML but the Wolfram Functions site does not. In the second experiment with ACL-ARC data, we compared our model with SnuggleTeX. Table 3 lists the various data statistics.

Table 3: Data statistic

Category	**No. of math expressions**
Bessel-TypeFunctions	1,960
Constants	709
ElementaryFunctions	37,965
GammaBetaErf	2,895
IntegerFunctions	1,612
Polynomials	1,489
ACL-ARC (ACL papers)	2,065

4.2 Evaluation Methodology

Given a Presentation MathML expression e, we assume that tree A is the correct Content MathML tree of expression e and tree B is the output of the automatic translation. The basic idea to evaluate the correctness of tree B is directly comparing it with tree A. In the experiments, we extended the conventional definition of "Translation Error Rate" and used a metric which is a combined version of

- the Tree Edit Distance [9]: the tree edit distance is the minimal cost to transform A into B using edit operations. There are three types of edit operation: substituting a node, inserting a node, and deleting a node.
- the Translation Error Rate [8]: the translation error rate is an error metric for machine translation that measures the number of edits required to change a system output into one of the references.

We called the new metric the Tree Edit Distance Rate (TEDR). TEDR is defined as the ratio of (1) the minimal cost to transform a tree A into another tree B using edit operations and (2) the maximum number of nodes of A and B. It can be computed using Eq. 1.

$$TEDR(A \rightarrow B) = \frac{TED(A,B)}{max\{|A|,|B|\}} \quad (1)$$

For example, the output tree of the expression $arctan(0) = 0$ is depicted in Figure 5. Compared with the reference tree in Figure 2, we need to substitute 1 nodes, insert 3 nodes, and delete 0 nodes, so that $TED(A,B) = 4$, while the maximum number of nodes of the two trees is 8. Therefore, $TEDR(A \rightarrow B) = \frac{4}{8} = 0.5$.

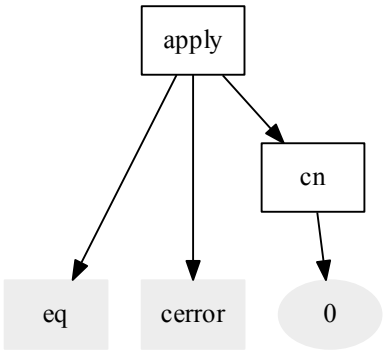

Figure 5: An example output tree of the expression $arctan(0) = 0$

4.3 Results Summary

It appeared that SnuggleTeX was not applicable to the data from the Wolfram Function site since it uses ASCII MathML but the site does not. Therefore, we could not do a comparison on this data. Our experimental results show that our approach gives reasonable results, that is, a 20 percent TEDR with large training data. For small data (less than 3000 training samples), the results vary from 50 to 75 percent TEDR.

For ACL-ARC data, the experimental results show that our system significantly outperforms SnuggleTeX in terms of the Tree Edit Distance Rate. Our system had a 24 percent lower TEDR in comparison with SnuggleTeX.

To investigate the correlation between the TEDR score and training set size, we set up an experiment using mathematical expressions in the Elementary Functions category. We started with one fifth of the data and increased the data by one fifth in each run. Our experimental results conformed with the theoretical analysis that the more training data we have, the better the results are.

4.4 Results Details

Table 4 and Table 5 show the TEDR of our method on the Wolfram Functions Site data and in comparison with SnuggleTeX on ACL ARC data, respectively.

Table 4: Results on the Wolfram Function Site data

Category	**Avg. No. of FR**	**Avg. No. of TR**	**TEDR**
Bessel-TypeFunctions	219	3,175	49.36
Constants	210	875	74.22
ElementaryFunctions	635	20,389	**20.89**
GammaBetaErf	421	4,848	59.89
IntegerFunctions	343	2,368	58.57
Polynomials	275	2,598	67.18

Table 5: Results on ACL-ARC data

Methods	**TED**	**Total Nodes**	**TEDR**
Our approach	17,562	26,085	**67.33**
SnuggleTeX	23,820	26,085	91.32

Table 6 and Figure 6 shows the correlation between TEDR score and training set size.

Table 6: Results on Elementary Functions category with different data size

Avg. TED	**Avg. total Nodes**	**TEDR**
31,105	47,351	65.69
43,985	114,413	38.44
55,601	180,305	30.84
62,681	248,464	25.23
66,486	318,282	20.89

5. Conclusions

We discussed the problem of semantic enrichment of mathematical expressions. Our experimental results show that our approach based on the statistical machine translation method for translating a Presentation MathML expressions to Content MathML expressions is a significant improvement over prior systems.

As we mentioned before, mathematical notations are context-dependent. That means we need to consider not only surrounding expressions but also the document that contains the notations in order to generate the correct semantic output. In the scope of this paper, we only considered the first sort of context information. Since this is a first attempt to translate from Presentation to Content MathML using a machine learning method, there is room for further improvement. Possible improvements are

- Increasing the training data so the system can cover more mathematical notations
- Expanding the work by incorporating the surrounding information of mathematical expressions, for example, definitions or other mathematical expressions.

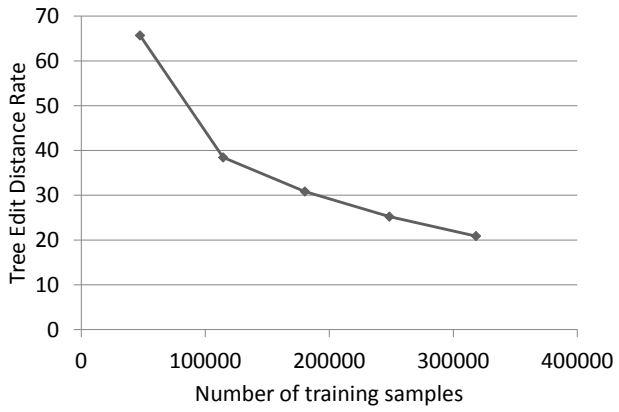

Figure 6: Correlation between TEDR score and training set size

Our approach combining automatic extraction of fragment rules and translation rules has shown promising results. The experimental results confirm that it would be helpful for automatic understanding of mathematical expressions. However, this is only a first step; many important issues remain for future studies. Currently, our system deals with a limited range of mathematical notations. In the future, we should consider expanding it to cover all mathematical notations.

References

[1] Franz Josef Och, Hermann Ney. "A Systematic Comparison of Various Statistical Alignment Models", Computational Linguistics, volume 29, number 1, pp. 19-51 March 2003.

[2] Masakazu Suzuki, Toshihiro Kanahori, Nobuyuki Ohtake and Katsuhito Yamaguchi. "An integrated OCR software for mathematical documents and its output with accessibility", ICCHP '04, LNCS, vol. 3118, pp 648–655, 2004.

[3] Mihai Grigore, Magdalena Wolska and Michael Kohlhase. "Towards Context-Based Disambiguation of Mathematical Expressions", The Joint Conference of ASCM 2009 and MACIS 2009: Asian Symposium on Computer Mathematics and Mathematical Aspects of Computer and Information Sciences, pp. 262-271, December 2009.

[4] World Wide Web Consortium. "Mathematical markup language", http://www.w3.org/Math/, 2011.

[5] David McKain. "SnuggleTeX", http://www2.ph.ed.ac.uk/snuggletex/, 2012.

[6] Deyan Ginev, Constantin Jucovschi, Stefan Anca, Mihai Grigore, Catalin David and Michael Kohlhase. "An Architecture for Linguistic and Semantic Analysis on the arXMLiv Corpus", Applications of Semantic Technologies (AST) Workshop at Informatik 2009, 2009.

[7] M.Q. Nghiem, K. Yokoi, Y. Matsubayashi, and A. Aizawa. "Mining coreference relations between formulas and text using Wikipedia", 2nd Workshop on NLP Challenges in the Information Explosion Era (NLPIX), pages 69-74, 2010.

[8] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul, "A Study of Translation Edit Rate with Targeted Human Annotation", Proceedings of Association for Machine Translation in the Americas, 2006.

[9] K. Zhang and D. Shasha, "Simple fast algorithms for the editing distance between trees and related problems", SIAM Journal on Computing, 1989.

[10] OpenOffice.org Math, http://www.openoffice.org/product/math.html, 2012

[11] ASCII MathML, http://www1.chapman.edu/jipsen/mathml/asciimath.html, 2012

[12] OpenMath, http://www.openmath.org/, 2012

[13] Kernighan, Brian W. and Cherry, Lorinda L., "A System for Typesetting Mathematics", Communications of the ACM (18), pages 151157, 1975.

[14] The Wolfram Functions Site, http://functions.wolfram.com/

[15] The Archives of the Association for Computational Linguistics, http://www.aclweb.org/archive/