Semantic Enrichment of Mathematical Expressions

for Mathematical Search

Minh-Quoc NGHIEM

Department of Informatics

The Graduate University for Advanced Studies

A thesis submitted for the degree of

Doctor of Philosophy

I would like to dedicate this thesis to my loving family

Acknowledgements

Foremost, I would like to thank my supervisor, Prof. Aizawa Akiko, who shared with me a lot of her expertise and research insight. I also like to express my gratitude to Prof. Miyao Yusuke, whose thoughtful advise often served to give me a sense of direction during my studies. I am deeply grateful to the Vietnam government and Japan Student Services Organization for the support that they gave me in order to study in Japan. And I wish to thank everybody in my laboratory for their advises and supports.

Abstract

Semantic enrichment of mathematical expressions is an important component in the *mathematics understanding system*, and plays a key role in *content-based search engine for mathematical expressions*. This dissertation presents a new approach to the semantic enrichment of mathematical expression problem. The problem is formulated as adding semantic form to the presentation form of mathematical expressions. More specific, it is the problem of translating a mathematical expression from its Presentation MathML to Content MathML.

The proposed approach uses statistical machine translation method to learn the translation rules automatically from parallel MathML markup data. The structural difference between Presentation and Content MathML is solved by introducing new segmentation rule. An enhancement to statistical machine translation system is made by using an support vector machines classifier to disambiguate the ambiguous mathematical terms with features extracted from both presentation form mathematical expressions and surrounding text. Combining theses system archives improvements over prior semantic enrichment systems.

This dissertation also presents a content-based mathematical search system which is an application of *semantic enrichment of mathematical expressions*. The approach uses *semantic enrichment of mathematical expressions* to convert mathematical expressions into their

content forms and searching is done using these content-based expressions. By considering the meaning of mathematical expressions, the quality of search system is improved over presentation-based systems.

This dissertation makes noteworthy contributions to mathematical-related research field. It confirms that natural language processing techniques can be applied to solve mathematical expressions related problems. Since mathematical content is a valuable information source for many users, this finding has important implications for developing mathematical-related systems.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The issue of *retrieving semantically similar mathematical expressions* has received considerable critical attention [Aizawa *et al.*, 2013]. Semantic search for mathematical expressions improves search accuracy by ascertaining the contextual meaning of mathematical terms as they appear in a search-able database to generate more relevant results. *Semantic enrichment of mathematical expressions* is an important component of the *mathematics understanding system*. It plays a key role in a *content-based search engine for mathematical expressions*. The process of associating semantic tags, usually concepts, with mathematical expressions, so-called *semantic enrichment*, is an important technology for fulfilling the dream of a global digital mathematical library.

A considerable amount of literature has been published on *retrieving semantically similar mathematical expressions* [Adeel *et al.*, 2008; Kohlhase & Prodescu, 2013; Kohlhase & Sucan, 2006; Liska *et al.*, 2013; Nguyen *et al.*, 2012; Wolfram, 2013]. These studies used two freely available toolkits, SnuggleTeX [McKain, 2013] and LaTeXML [Miller, 2013], for *semantic enrichment of mathematical expressions*. Mathematical expressions are first semantically enriched to their content-based format, which is Content MathML (see 2.1.4). Search is then per-

formed by matching the queries with indexed mathematical terms in the database. The experimentally obtained results show that the proposed approaches, using classical information retrieval strategies, can perform better than other methods.

Research efforts to date have tended to examine *retrieval of semantically similar mathematical expressions* specifically rather than *semantic enrichment of mathematical expressions*. Much uncertainty remains about the relation between the *performance of mathematical search systems* and the *performance of semantic enrichment components*. Prior attempts to address the *semantic enrichment of mathematical expressions* problem include SnuggleTeX [McKain, 2013] and LaTeXML [Miller, 2013]. These systems use handwritten rule-based methods for disambiguation and translation. Two issues limit these solutions:

- As handwritten rule-based systems, these systems require mathematical knowledge and human involvement;
- These systems remain at the experimental stage because of difficulties with processing complex mathematical symbols and because of the wide-ranging nature of mathematical expressions.

Furthermore, no research has been found that has surveyed how well semantic enrichment components perform.

Therefore, this study has two primary aims:

- To develop an understanding of *semantic enrichment of mathematical expressions* and its performance.
- To investigate the contribution of *semantic enrichment of mathematical expressions* to *content-based mathematical search systems*.

This dissertation will first develop a *semantic enrichment of mathematical expressions system* based on machine-learning techniques and will then go on to

establish a method for automatic evaluation of the performance of *semantic enrichment of mathematical expressions system*. This dissertation will also develop a *content-based mathematical search system* and examine the contribution of *semantic enrichment of mathematical expressions* to a *content-based mathematical search system*.

This study was undertaken to address the following research questions:

- How well do machine-learning-based approaches perform compared to hand-written rule-based approaches on the problem of *semantic enrichment of mathematical expressions*?
- To what extent can we apply natural language processing techniques to the problem of *semantic enrichment of mathematical expressions*?
- How is the performance of *semantic enrichment of mathematical expressions system* affecting *content-based mathematical search systems*?

Chapter 3, 4, and 5 of this dissertation respectively address the three research questions above.

This dissertation follows machine-learning-based approaches, with in-depth analysis of the relation between mathematical expressions and natural language text. The data used in this dissertation were collected from the Wolfram Functions Site [Wolfram, 2013], the world's largest collection of mathematical expressions. Other data were also prepared by annotating mathematical expressions drawn from 20 papers from the archives of the Association for Computational Linguistics Anthology Reference Corpus [Bird *et al.*, 2008; Kan, 2013]. Experiments are performed using a ten-fold cross validation or reserved test set on numerous mathematical expressions to ensure the correctness of the proposed approaches.

This report is the first of a study making a complete evaluation of a *semantic enrichment of mathematical expressions* system. This study provides a testing dataset for mathematical term sense disambiguation and suggests two standard metrics for evaluating this task: the tree-edit-distance error rate and the perfect translation rate. This is also the first study to evaluate the relation between the *performance of mathematical search system* and the *performance of semantic enrichment component*. The results of this research will serve as a base for future studies in this field.

Throughout this dissertation, the term "semantic enrichment" of mathematical expressions will be used to refer to *adding Content MathML markup to Presentation MathML markup of mathematical expressions*. This definition is the same as that of David McKain [McKain, 2013] who defined "semantic enrichment" as generating "semantically richer" outputs than its usual display-oriented Presentation MathML. Fundamentally, semantic enrichment means enriching the content of data by tagging, categorizing, or classifying data in relation mutually, to dictionaries, or other base reference sources. A considerable amount of literature has been published on "semantic enrichment" including semantic enrichment of journal articles [Batchelor & Corbett, 2007], and semantic enrichment of text [Peñas & Hovy, 2010].

The main reason for choosing this topic is inspired by the idea of Tim Berners-Lee about "Semantic Web" [Berners-Lee, 2000]. His vision suggests that computers of the future can analyze all data on the Web, as well as understand the contents of human language. With the huge amount of information available on the Web, automatic extraction, organization, and summarization of web information is necessary. In the future, it will not be necessary for people to surf hundreds of web pages, or read dozens of articles to find the information they need. A computer will automatically do all the work: analyze an article, find

related articles, and summarize all the information, and then provide it to users. Semantic enrichment of mathematical expressions is the first step toward this goal.

The overall structure of the dissertation takes the form of six chapters, including this introductory chapter, as depicted in Figure 1.1. Chapter 2 begins to lay out the background by introducing mathematical expression representation. It then provides an overview about the related work on *semantic enrichment of mathematical expressions* and *mathematical search systems*. Chapter 3 presents a method for *semantic enrichment of mathematical expressions* based on a Statistical Machine Translation approach. Portions of this chapter were published previously as [Nghiem *et al.*, 2013a]. Chapter 4 describes an additional enhancement for *semantic enrichment of mathematical expressions* by introducing the method for sense disambiguation of mathematical terms. Portions of this chapter were published previously as [Nghiem *et al.*, 2013b,c]. Chapter 5 describes a method for *content-based mathematical search system* and the contribution of *semantic enrichment of mathematical expressions* to that system. Chapter 6 concludes the dissertation and points to possible avenues for future work.

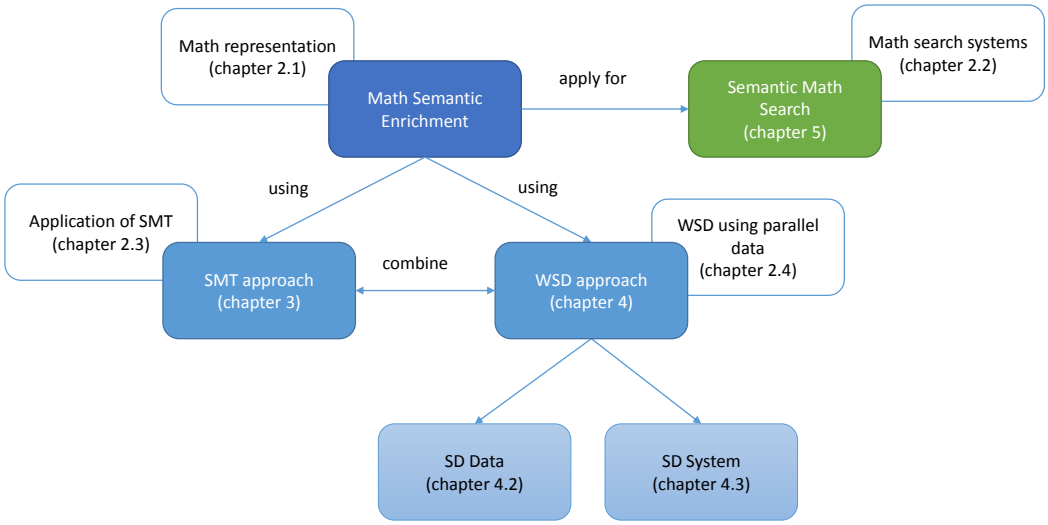

Figure 1.1: Overview structure of the thesis.

Chapter 2

Literature review

This chapter begins by laying out the background by introducing representation of mathematical expressions. It then provides an overview about the related work on *semantic enrichment of mathematical expressions* and *mathematical search system*.

2.1 Mathematical expression representation

Many websites such as Wikipedia use images to present mathematical expressions. Using images, the rendering of mathematical expression is independent of client-side browser resources. Figure 2.1 portrays a page from a mathematical section on Wikipedia with mathematical expressions displayed as images. Images provide a methodologically uniform approach, but the result is not machine-readable. It is possible to use optical character recognition (OCR) software to recognize mathematical expressions and convert them to accessible formats. A well-known example of mathematical OCR software is InftyReader [Suzuki *et al.*, 2004].

Several markup languages for mathematical expression representation. Common mathematical markup languages are TEX, ASCIIMathML, OpenMath, OM-

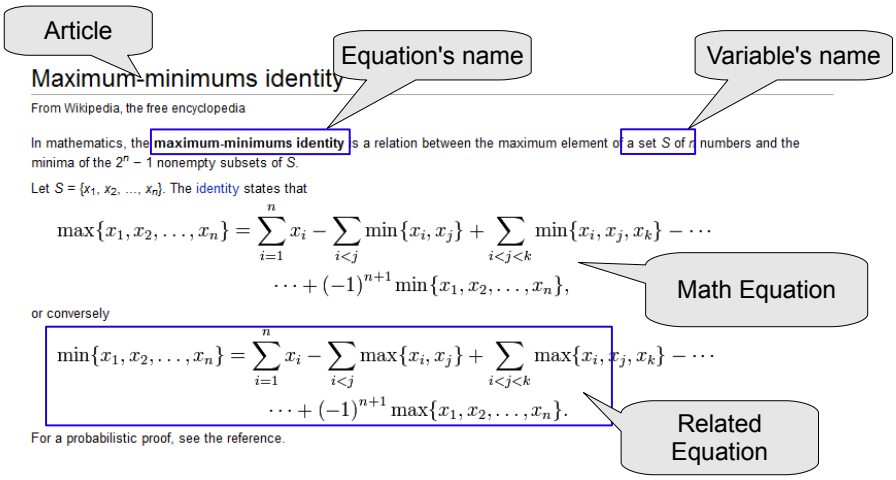

Figure 2.1: Mathematical expressions on Wikipedia.

Doc, and MathML. The markup languages are divisible into two main types: presentation markup and content markup. Presentation markups, which include TEX, ASCIIMathML, and Presentation MathML, are used to describe the layout structure of a mathematical expression. Content markups, which include OpenMath, OMDoc, and Content MathML, provide explicit encoding of the underlying mathematical structure of an expression.

2.1.1 TEX

TEX [Knuth, 1984; Lamport, 1986] is a typesetting system which can typeset complex mathematical expressions. TEX is popular in academia and has been commonly used by many researchers, especially in mathematics. TEX provides a text syntax for mathematical expressions so that authors can typeset expressions in their papers by themselves. An expression is printed as a person would write it by hand, or as a person would typeset the expression. On certain web pages, such as Wikipedia, mathematical expressions can be displayed in TEX format using the *alt* attribute.

Figure 2.2 portrays a mathematical expression and its written form using TEX.

```
\sin \alpha = \frac {\textrm{opposite}} {\textrm{hypotenuse}} = \frac {a} {h}
```

$$\sin \alpha = \frac{opposite}{hypotenuse} = \frac{a}{h}$$

Figure 2.2: Mathematical expression written using TEX.

2.1.2 OpenMath

OpenMath [Buswell *et al.*, 2004] is a standard markup language for representing the meaning of mathematical expressions. It emphasizes the representation of semantic information and is not intended to be used directly for presentation. OpenMath uses an extensible Content Dictionary for encoding the semantics of mathematics. It enables mathematical expressions to be exchanged among computer programs, stored in databases, or published on the world wide web. OpenMath has a strong relation to the MathML markup language; it is useful to complement MathML.

2.1.3 OMDoc

OMDoc [Kohlhase, 2006] (Open Mathematical Documents) is a semantic markup format for mathematical documents. It covers the whole range of written mathematics including mathematical expressions, statements, and theories. OMDoc is used in e-learning, data exchange, and document preparation. It also emphasizes representation of semantic information that is not primarily presentation-oriented. OMDoc uses OpenMath and Content MathML formats to present mathematical expressions.

2.1.4 MathML

The best-known open markup format for representing mathematical expressions on the web is MathML [Ausbrooks *et al.*, 2010]. It is a format recommended by the W3C Math Working Group as a standard to represent mathematical expressions. MathML is an XML application for describing mathematical notations and encoding mathematical content within a text format. MathML has encoding of two types: content-based encoding, called Content MathML and presentation-based encoding, called Presentation MathML. Content MathML addresses the semantic meaning whereas Presentation MathML addresses the display of the mathematical expressions.

Presentation MathML specifically examines the display of an expression and has about 30 elements. The presentation elements of Presentation MathML are divided into two classes: token elements and layout schemata. Token elements represent identifier names, function names, numbers, and so forth. Layout schemata build expressions from parts. Figure 2.3 shows the Presentation Markup of the expression `arctan(0)=0` [1].

Content MathML provides an explicit encoding of the underlying mathematical meaning of the mathematical expression rather than its layout. It uses the *apply* element to represent the function application. The function being applied is the first child element under *apply*, remaining child elements are operands or parameters. Content MathML has over a hundred different elements for different functions and operators. Figure 2.4 shows the Content Markup of the expression `arctan(0)=0`.

One disadvantage of MathML is it is not designed to be written by a human. To overcome this problem, ASCIIMathML [ASCIIMathML, 2013] provides an easy means to write mathematical expressions. Mathematical expressions repre-

[1] http://functions.wolfram.com/01.14.03.0001.01

Presentation MathML

```
<mrow>
   <mrow>
      <msup>
         <mi>tan</mi>
         <mrow>
            <mo>-</mo>
            <mn>1</mn>
         </mrow>
      </msup>
      <mo>(</mo>
      <mn>0</mn>
      <mo>)</mo>
   </mrow>
   <mo>=</mo>
   <mn>0</mn>
</mrow>
```

Figure 2.3: MathML presentation markup for the expression `arctan(0)=0`.

sented using ASCIIMath markup are easy to produce because they mainly use ASCII characters to represent the mathematical symbols. The script of ASCIIMathML is open source and available under a GPL license. Figure 2.5 shows the ASCIIMathML markup of the well-known quadratic formula.

MathML presentation and content markups are chosen in this dissertation to represent mathematical expressions for the following reasons:

ContentMathML

```
<apply>
   <eq/>
   <apply>
      <arctan/>
      <cn>0</cn>
   </apply>
   <cn>0</cn>
</apply>
```

Figure 2.4: MathML content markup for the expression `arctan(0)=0`.

ASCIIMathML

```
x=(-b+-sqrt(b^2-4a c))/(2a)
```

Figure 2.5: Well-known quadratic formula written in ASCIIMathML.

- Since its release in 1997, MathML has grown to become a general format that enables mathematics to be served, received, and processed in widely various applications.
- MathML is useful to encode both mathematical notation and mathematical content.
- Large collections of mathematical expressions are already available in MathML, and access to these collections is easy.
- All other markups including eqn [Kernighan & Cherry, 1975], OpenOffice.org Math [Oracle, 2013], ASCIIMathML, and OpenMath can be converted into MathML using freely available toolkits [Stamerjohanns *et al.*, 2009].

2.2 Mathematical Presentation to Content Conversion

As described in the previous section, mathematical expressions consist of presentation-based and content-based formats. *Mathematical search systems* can either use presentation-based or content-based format as indexed terms. Whereas presentation-based systems capable of returning exact or similar matches, content-based systems can return more meaningful results. However, content-based search systems need a mathematical expression in content-based format. Because content-based

mathematical expressions are usually not available, conversion or *semantic enrichment of mathematical expressions* is necessary.

Few studies have addressed the problem of *semantic enrichment of mathematical expressions*. In this section, we list some of the work on interpreting the meaning of mathematical expressions. Two available systems support generation of Content MathML markup from Presentation MathML or LaTeX: SnuggleTeX and LaTeXML. Other systems of Grigore *et al.* [2009] and Wolska & Grigore [2010]; Wolska *et al.* [2011] specifically identify the correct meaning or class of a mathematical term.

2.2.1 System of Grigole et al.

Grigore *et al.* [2009] proposed an approach to understanding mathematical expressions based on the text surrounding the mathematical expressions. The main concept underlying this approach is to use the surrounding text for disambiguation based on word sense disambiguation and lexical similarities. 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). The highest semantic 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. When this approach was evaluated for 451 manually annotated mathematical expressions, the best result was an $F_{0.5}$ score of 68.26.

2.2.2 SnuggleTeX

A project called SnuggleTeX [McKain, 2013] addresses the semantic interpretation of mathematical expressions. The project provides a direct method to generate Content MathML from Presentation MathML based on manually encoded

rules. The current version at the time of writing this paper supports operators that are the same as ASCIIMathML [ASCIIMathML, 2013]. For example, it uses the ASCII string "\in" instead of the symbol "$\in$". One important shortcoming of this approach is that it always makes the same interpretation for the same Presentation MathML element.

2.2.3 LaTeXML

Lamapun [Ginev *et al.*, 2009] project investigates semantic enrichment, structural semantics, and ambiguity resolution in mathematical corpora. The project uses LaTeXML [Miller, 2013] for conversion from LaTeX to MathML. Unfortunately, no evaluation of these systems has been made to date.

2.2.4 System of Wolska et al.

Wolska & Grigore [2010]; Wolska *et al.* [2011] presented a knowledge-poor method for identifying the denotation of simple symbolic expressions in mathematical discourse. Based on statistical co-occurrence measures, the system sorted a simple symbolic expression under one of seven predefined concepts. Here, the authors found that lexical information from the linguistic context immediately surrounding the expression improved results. This approach achieves 66% agreement with the gold standard of manual annotation by experts. From our perspective, the predefined concepts are closely related to syntactic function, not the semantics of the terms.

2.3 Statistical Machine Translation and its applications

Statistical machine translation (SMT) [Brown *et al.*, 1990, 1993; Chiang, 2005; Yamada & Knight, 2001] is by far the most widely studied machine translation method. SMT uses a very large dataset of good translations, which comprises a corpus of texts already translated into another language called a parallel corpus. It uses these parallel texts to infer a statistical model of translation automatically. The statistical model is then applied to new texts to derive a translation.

2.3.1 Word alignment

Most of the methods on extracting translation rules from a parallel corpus start with a word alignment. In the word alignment process, each element in one language is matched with the corresponding element in the other language. Many word aligners are available today, including GIZA++ [Och & Ney, 2003], MGIZA++ [Gao & Vogel, 2008], and Berkeley Aligner [Liang *et al.*, 2006]. This dissertation uses GIZA++ toolkit which implements the IBM Models 1–5 and an HMM word alignment model. The training of these models is done using the Expectation-Maximization (EM) algorithm.

EM algorithm is used to estimate parameters in statistical models, where the model depends on unobserved variables. In word alignment, the EM algorithm first initializes model parameters. It then iterates alternates between performing an expectation step, and a maximization step. Expectation step assigns probabilities to the missing data. Maximization step estimates model parameters from completed data. Figure 2.6 presents an example of EM algorithm with source language is Presentation MathML elements and target language is Content MathML elements.

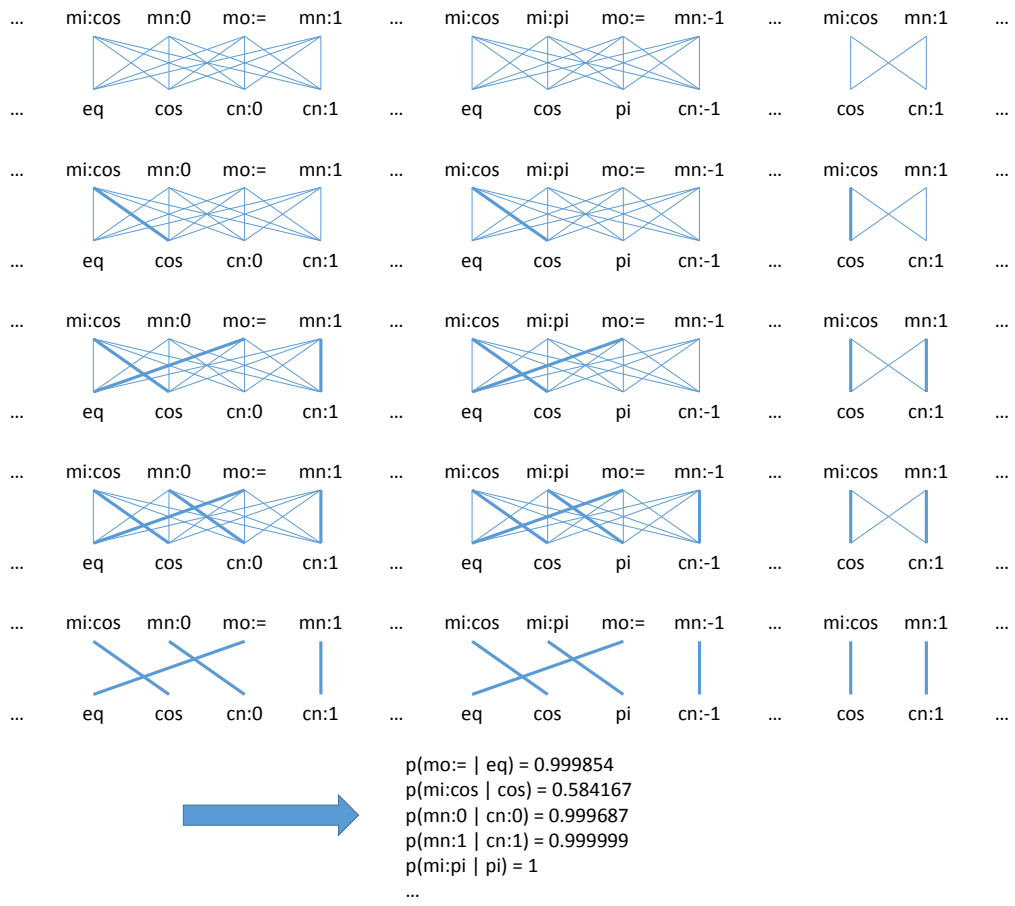

Figure 2.6: Illustration of EM algorithm.

2.3.2 Applications of Machine Translation

Word Alignment-based Semantic Parsing [Wong & Mooney, 2006] applies MT techniques to learn semantic parsers. The fundamental idea is to use SMT to learn to translate from natural language to a meaning representation language. A word alignment model is used for lexical acquisition, and a syntax-based translation model is used as the parsing model. Results of this study show that SMT is applicable to semantic parsing.

Several reports have described encouraging results for word sense disambiguation based on parallel corpora [Carpuat & Wu, 2007; Chan & Ng, 2005; Diab &

Resnik, 2002; Lefever & Hoste, 2010; Lefever *et al.*, 2011; Padó & Lapata, 2009; Tufiş *et al.*, 2004]. Ide *et al.* [2002] used translation equivalents derived from parallel aligned corpora to determine sense distinctions that are applicable to automatic sense-tagging. They evaluated their work using a subset of 33 nouns covering a range of occurrence frequencies and degrees of ambiguity [Ide *et al.*, 2001], with results indicating no significant difference in agreement rates for the algorithm and for human annotators. The main limitation of this study is its dependence on aligned corpora, which are not easily obtainable.

2.4 Mathematical Search System

As the demand for mathematical searching increases, several mathematical retrieval systems have come into use. Most systems use the conventional text search techniques to develop a new mathematical search system [National Institute of Standards and Technology, 2013; Springer, 2013; Uniquation, 2013]. Some systems with specific format for mathematical content and queries [Altamimi & Youssef, 2008; Miner & Munavalli, 2007; Yokoi & Aizawa, 2009; Youssef, 2005; Youssef & Altamimi, 2007]. Based on a different mathematical markup, current mathematical search systems are divisible into presentation-based and content-based systems. Presentation-based systems duel with the presentation form whereas content-based systems duel with the meanings of mathematical formulae.

2.4.1 Presentation-based systems

2.4.1.1 Springer LaTeXSearch

Springer offers a free service, Springer LaTeX Search [Springer, 2013], to search for LaTeX code within scientific publications. It enables users to locate and view

equations containing specific LaTeX code, or equations containing LaTeX code that is similar to another LaTeX string. A similar search in Springer LaTeX Search ranks the results by measuring the number of changes between a query and the retrieved formulae. Each result contains an entire LaTeX string, a converted image of the equation, and information about and links to its source. However, the ranking algorithm performs poorly when only measuring the number of changes.

2.4.1.2 MathFind

MathFind [Munavalli & Miner, 2006] is a math-aware search engine under development by Design Science. This work extends the capability of existing text search engines to search mathematical content. The system analyzes expressions in MathML and decomposes the mathematical expression into a sequence of text-encoded math fragments. Queries are also converted to sequences of text and the search was done as normal text search. However, treating mathematical expressions as text can not fully capture the structural notations of mathematical formulae.

2.4.1.3 Digital Library of Mathematical Functions

The Digital Library of Mathematical functions (DLMF) project at NIST is a mathematical database available on the Web [Miller & Youssef, 2003; National Institute of Standards and Technology, 2013]. Two approaches are used for searching for mathematical formulas in DLMF. The first approach converts all mathematical content to a standard format. The second approach exploits the ranking and hit-description methods. These approaches enable simultaneous searching for normal text as well as mathematical content.

In the first approach [Youssef, 2005], they propose a textual language, Textualization, Serialization and Normalization (TexSN). TeXSN is defined to normalize

non-textual content mathematical content to standard forms. User queries are also converted to the TexSN language before processing. Then, a search is performed to find the mathematical expressions that match the query exactly. As a result, similar mathematical formulae are not retrieved.

In the second approach [Youssef, 2007], the search system treats mathematical expressions as a document containing a set of mathematical terms. This paper introduces new relevance ranking metrics and hit-description generation techniques. They claimed that the new relevance metrics are far superior to the conventional tf-idf metric. The new hit-descriptions are also more query-relevant and representative of the hit targets than conventional methods. However, this paper lacked a thorough subjective evaluation including numerous users and a carefully selected benchmark of queries.

Other notable math search systems include Math Indexer and Searcher Sojka & Líška [2011], EgoMath Miutka & Galambo [2011], and ActiveMath Siekmann [(visited on 01 March. 2014].

2.4.2 Content-based systems

2.4.2.1 Wolfram Function

The Wolfram Functions Site [Wolfram, 2013] is the world's largest collection of mathematical formulas accessible on the Web. Currently the site has 14 function categories containing more than three hundred thousand mathematical formulae. This site allows users to search for mathematical formulae from its database. The Wolfram Functions Site proposes similarity search methods based on MathML. However, content-based search is only available with a number of predefined constants, operations, and function names.

2.4.2.2 MathWebSearch

The MathWebSearch system [Kohlhase & Prodescu, 2013; Kohlhase & Sucan, 2006] is a content-based search engine for mathematical formulae. It uses a term indexing technique derived from an automated theorem proving to index Content MathML formulae. The system first converts all mathematical formulae to Content MathML markup and uses substitution-tree indexing to build the index. The authors claimed that search times are fast and unchanged by the increase in index size. However, MathWebSearch is currently restricted to an exact formula search without similarity search and full-text search.

2.4.2.3 MathGO!

Adeel *et al.* [2008] proposed a mathematical search system called the MathGO! Search System. The approach used conventional search systems using regular expressions to generate keywords. For better retrieval, the system clustered mathematical formula content using K-Som, K-Means, and AHC. They did experiments on a collection of 500 mathematical documents and achieved around 70–100 percent precision. However, the complexity of their algorithm is expected to increase when the number of templates increases.

2.4.2.4 MathDA

Yokoi and Aizawa [Yokoi & Aizawa, 2009] proposed a similarity search method for mathematical expressions that is adapted specifically to the tree structures expressed by MathML. They introduced a similarity measure based on Subpath Set and proposed a MathML conversion that is apt for it. Their experiment results showed that the proposed scheme can provide a flexible interface for searching for mathematical expressions on the Web. However, the similarity calculation is the bottleneck of the search when the database size increases. Another shortcoming

of this approach is that the system only recognizes symbols and does not perceive the actual values or strings assigned to them.

2.4.2.5 System of Nguyen et al.

Nguyen *et al.* [2012] proposed a math-aware search engine that can handle both textual keywords and mathematical expressions. They used Finite State Machine model for feature extraction and representation framework captures the semantics of mathematical expressions For ranking, they used the passive–aggressive online learning binary classifier. Evaluation was done using 31,288 mathematical questions and answers downloaded from Math Overflow [MathOverflow, 2013]. Experimental results showed that their proposed approach can perform better than baseline methods by 9%. However, results for other kinds of mathematical document retrieval have not been reported.

Chapter 3

Semantic Enrichment of mathematical expressions

This chapter presents a method for *semantic enrichment of mathematical expressions* based on Statistical Machine Translation approach. Portions of this chapter were previously published as [Nghiem *et al.*, 2013a].

3.1 Overview

The semantic enrichment of mathematical expressions is among the most significant areas of discussion related to the digitization of mathematical and scientific content and its applications. The challenge entails associating semantic tags, usually concepts, with mathematical expressions. Encoding the underlying mathematical meaning of an expression confers several benefits:

- It facilitates more precise information exchange between systems that process mathematical objects;
- It improves the accuracy of mathematical search systems by enabling semantic searching of mathematical expressions;

- It also benefits computer algebra systems, automated reasoning systems, and multi-lingual translation systems.

As with natural language, the semantic enrichment of mathematical expressions is a nontrivial task. Although more rigorous than natural language, mathematical notations are ambiguous, context-dependent, and vary from community to community. The difficulty in inferring semantics from a presentation stems from the many-to-many potential mappings from presentation to semantic [Ausbrooks *et al.*, 2010]. Examples include binomial coefficients, which can be presented in varying notations: $C(n,k)$, ${}_nC_k$, ${}^nC_k$, C_n^k, C_k^n. Moreover, each notation can have other author-dependent meanings aside from the binomial coefficient itself.

This chapter introduces an automatic semantic enrichment method for mathematical expressions to analyze and disambiguate mathematical terms. In this study, MathML [Ausbrooks *et al.*, 2010] Presentation Markup is used to display mathematical expressions and MathML Content Markup is used to convey mathematical meaning. The semantic enrichment task then becomes the task of generating Content MathML outputs from Presentation MathML expressions.

Prior attempts to address this problem include SnuggleTeX [McKain, 2013] and LaTeXML [Miller, 2013]. These systems use handwritten rule-based methods for disambiguation and translation. The first discussions and analyses of *semantic enrichment of mathematical expressions* emerged during version 1.2.0 of SnuggleTeX within the Maths Assess Project [MathsAssess, 2013]. However, these features are still to be considered experimental and no research has been found that surveyed how well semantic enrichment component performs.

This chapter proposes an approach based on Statistical Machine Translation (SMT) [Brown *et al.*, 1990] techniques. In the proposed framework, 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 presentation expression p. The probability distribution is automatically calculated given parallel markup MathML data which contains both Presentation and Content MathML markup for a single expression. The data used in this study was collected from the Wolfram Functions Site [Wolfram, 2013] (WFS). Another parallel markup MathML data was also prepared by annotating mathematical expressions drawn from 20 papers from the ACL Anthology Reference Corpus [Bird *et al.*, 2008; Kan, 2013] (ACL-ARC).

The SMT approach has a number of attractive features over rule-based approach. The first advantage of using SMT is we can make better use of resources. As mentioned above, there is a great deal of mathematical expressions in MathML parallel markup format, i.e. the Wolfram Functions Site data. The second advantage of using SMT is we can quickly produce the translation rule set. The rule-based translation system requires the manual development of rules, which can be costly, and which often do not generalizable.

The main disadvantage of the SMT method is that it does not work well between languages that have significantly different word orders which is the case of Presentation to Content MathML. Several attempts have been made to overcome the different word order problem by re-ordering one language side [Birch & Osborne, 2011; Nagata *et al.*, 2006; Yang *et al.*, 2012]. To deal with this shortcoming, the dissertation proposes a type of rule, named the "segmentation rule". Segmentation rules are used for the purpose of reducing the different word order between Presentation and Content MathML expressions. These rules are learned at the same time the system learns the translation rule from MathML parallel markup data.

Evaluation was performed by using a ten-fold cross validation on mathematical expressions from the six categories of the Wolfram Functions Site. This ex-

periment evaluated the effectiveness of the proposed learning method. Another experiment was performed to assess the correlation between systems performance and training set size. The proposed method is compared to prior work [McKain, 2013] using a data set collected from ACL-ARC scientific papers. Results show that proposed approach yields improvements in the mathematics semantic enrichment problem, generating fewer errors and outperforming this previous work.

3.2 Proposed System

3.2.1 System Overview

The same framework of SMT system is applied here. The parallel markup expressions are used to automatically infer a statistical model of translation (rules for translation and their probabilities). The statistical model is then applied to new expressions to derive a translation. Figure 3.1 gives the system framework.

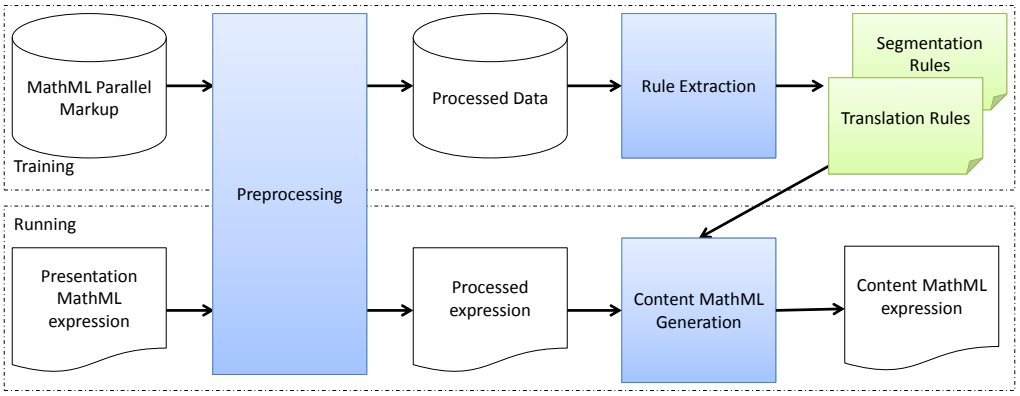

Figure 3.1: System Framework

The system has two phases, a training phase and a running phase, and consists of three main modules.

- Preprocessing: Processes MathML expressions. It removes error expressions and XML tags that convey no meaning.

- Rule Extraction: Extracts rules for translation, given the training data. There are two types of rules: segmentation rules and translation rules. Each rule is associated with its probability.
- Content MathML Generation: Generates Content MathML expressions from input Presentation MathML expressions using rules from the Rule Extraction step.

3.2.2 Preprocessing

In MathML Presentation Markup, certain elements are used for formatting purposes only: the *mtext* and *mspace* tags are used to insert a space between expressions. Some *mtable* tags are used to number the mathematical expressions. A pair of parentheses indicates that the expressions in the parentheses belong together. These elements are removed in specific cases where the structure encodes the same information. Keeping these elements can produce misleading results. Expressions with more than 200 nodes in their Content Markup are removed for simplification. Figure 3.2-(b) illustrates an example of this step.

The data contains expressions that convey the same meaning, but their Content MathML are written in different ways. To improve the alignment results, the system normalized two expressions having the same content meaning on the Content MathML side. Currently, these three cases are implemented in the proposed system:

- $sqrt(X)$ and $X^{\frac{1}{2}}$
- $X - Y$ and $X + (-Y)$
- $\frac{1}{X}$ and X^{-1}

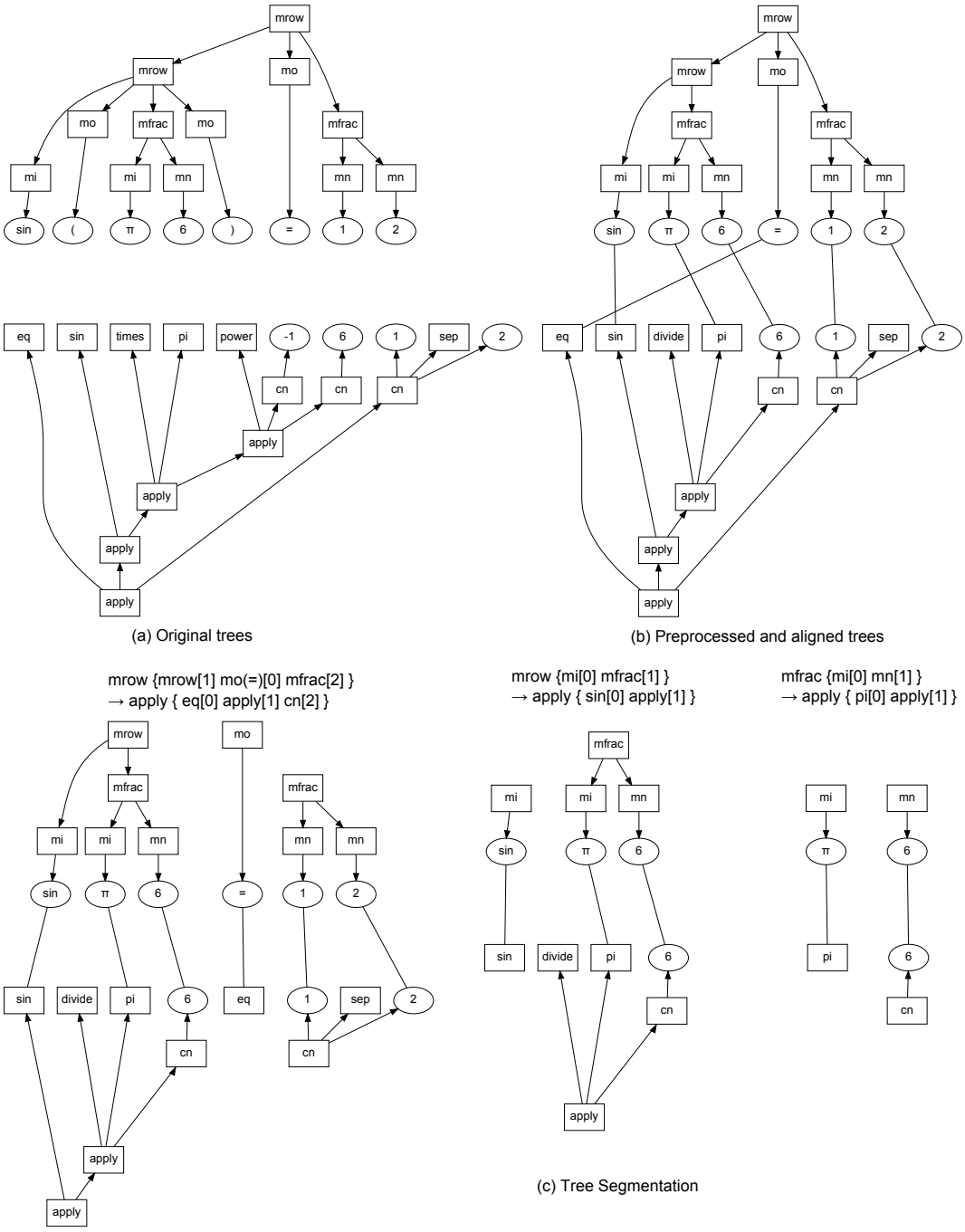

Figure 3.2: (a) Original Presentation and Content MathML Markup tree representations (b) preprocessed trees and the alignment between the nodes (c) segmentation process.

3.2.3 Extracting Rules

There are two sets of rules extracted: segmentation rules and translation rules. Segmentation rules are used to segment Presentation MathML trees into smaller subtrees. Translation rules are used to translate Presentation MathML trees into Content MathML trees. Segmentation rules and translation rules operate the same as "grammar rules" and "rule table" in SMT systems.

3.2.3.1 Extracting Segmentation Rules

Segmentation rules are proposed to divide a large Presentation MathML tree into smaller subtrees while maintaining alignment with their corresponding Content MathML trees.

Long sentences pose a common problem for SMT. System training with long sentence pairs requires more memory and CPU time. The translation quality is also low due to poorly aligned words in long sentence pairs. In this study, 151.2 nodes is the average length of mathematical expressions in the dataset (counting only the leaf nodes). The 30.66 average node is still high, even after removing expressions with more than 200 nodes in their Content Markup. Long mathematical expressions must be segmented into shorter ones. Note that segmenting MathML expressions is easier than segmenting natural language sentences since the structural information is explicitly encoded using XML.

For a given mathematical expression pair (p, c), we have $p_1, p_2, ..., p_n$ as subtrees of p and $c_1, c_2, ..., c_m$ as subtrees of c. A segmentation of (p, c) is defined as a sequence of subtree pairs $(p_{s_1}, c_1), (p_{s_2}, c_2), ..., (p_{s_m}, c_m)$, where $p_{s_1}, p_{s_2}, ..., p_{s_m}$ are corresponding subtrees of

$c_1, c_2, ..., c_m$.

To achieve segmentation, GIZA++ [Och & Ney, 2003] toolkit is used to obtain alignment between the leaf nodes of Presentation and Content MathML

trees. Figure 3.2-(b) shows an example of this alignment. Segmentation Rules are extracted based on this alignment. For each Content MathML subtree c_i, the corresponding Presentation MathML subtree p_{s_i} is the subtree satisfying the following condition:

$$p_{s_i} = \arg\max_j P(p_j|c_i, a) \tag{3.1}$$

$P(p_j|c_i, a)$ is calculated by obtaining the ratio of number of alignments between p_j and c_i to the total of alignment in a, where variable a represents the alignments between p and c.

$$P(p_j|c_i, a) = \frac{count[a(p_j, c_i)]}{|a|} \tag{3.2}$$

The following constraint is applied: distinct Presentation subtrees cannot be aligned with the same Content subtree. The only exception is the case of operators. Many identical operators subtrees in a Presentation subtree can be aligned with one Content subtree. This allowance is made because the Content function can have more than two arguments, while the Presentation operator permits only two. A segmentation that does not satisfy this constraint is invalid. A segmentation rule is created each time the system segments the tree. Each segmentation rule is associated with a probability which represents how likely it is that the right-hand side of the rule will happen given the left-hand side.

Figure 3.2-(c) shows an example of segmentation process and extracted segmentation rules. Table 3.1 gives examples of segmentation rules. In the table, the numbers, such as [1], represent corresponding Presentation and Content markup subtrees.

Table 3.1: Examples of segmentation rules extracted from Wolfram Functions Site dataset

Segmentation Rule	Probability
mrow { mrow[1] mo(=)[0] msup[2] } → apply { eq[0] apply[1] apply[2] }	1
mrow { mrow[1] mo(/;)[0] mrow[2] } → apply { ci(Condition)[0] apply[1] apply[2] }	0.9998
mrow { mrow[1] mo(=)[0] mrow[2] } → apply { eq[0] apply[1] apply[2] }	0.9946
mrow { mrow[1] mo($\propto$)[0] mrow[2] } → apply { ci(Proportional)[0] apply[1] apply[2] }	0.9511
mrow { msup[1] mo(.)[0] mrow[2] } → apply { times[0] apply[1] apply[2] }	0.8582

3.2.3.2 Extracting Translation Rules

If the subtree cannot be segmented or if the segmentation is invalid, a translation rule is extracted. Translation rules are used to translate a Presentation tree directly into a Content tree. Each translation rule is also associated with its frequency of occurrence throughout the training process. Training halts when no expressions can be segmented. Algorithm 1 gives the pseudo code for extracting the rules.

Function "UpdateProbability" uses Equation (3.2) to calculate the probability of each rule. Function "GetTranslationRule" and "GetSegmentationRule" extract the appropriate rules from the traning sample. Function "ExtractRule" calls ifself recursively until the subtree cannot be segmented anymore. Table 3.2 shows examples of translation rules.

Note that the rule <mo>-</mo> → <plus/> is a legal translation rule but its probability is low. The rule is extracted from those expressions which contain addition of 3 or more terms, i.e. $X - Y + Z$ (plus between X and $-Y$ and Z), these expressions were not normalized in the preprocessing step. Alignment errors or segmentation errors can also lead to wrong rule extraction.

Algorithm 1 Extract Translation Rules and Segmentation Rules

Input: set of training MathML files parallel markup M
Output: list of segmentation rules SR
list of translation rules TR
function EXTRACTRULES(M)
$SR \leftarrow \emptyset$
$TR \leftarrow \emptyset$
$A =$ ALIGN(M) ▷ alignments of nodes (output of GIZA++)
for all $m \in M$ **do**
EXTRACTRULE(m,A,SR,TR)
end forreturn SR, TR
end function

function EXTRACTRULE(m,A,SR,TR)
$tr =$ GETTRANSLATIONRULE(m) ▷ Extract the translation rule
if TR contains tr **then**
UPDATEPROBABILITY(TR)
else
$TR \leftarrow TR \cup \{tr\}$
end if
$sr =$ GETSEGMENTATIONRULE(m) ▷ Extract the segmentation rule
if SR contains sr **then**
UPDATEPROBABILITY(SR)
else
$SR \leftarrow SR \cup \{sr\}$
end if
let subTrees[1 .. N] be subtrees of m
for $i = 1 \rightarrow N$ **do**
EXTRACTRULE($subTrees[i]$,A,SR,TR)
▷ Extract rules of each subtree
end for
end function

function GETRULES(m) ▷ GetTranslationRule, GetSegmentationRule
for all $stP \in m$ **do**
$stC, count \leftarrow$ GETMAXALIGN(m,A) ▷ get the content sub-tree stC has most alignments to stP
end for
return MAKERULE($subTreeP$, $subTreeC$, $count$)
end function

Table 3.2: Examples of translation rules extracted from Wolfram Functions Site dataset

Translation Rule	**Probability**
< mo > . < /mo > → < times/ >	1
< mo > ∈ < /mo > → < in/ >	1
< mi > m < /mi > → < ci > m < /ci >	1
< mo > /; < /mo > → < ci > Condition < /ci >	0.9998
< mo > = < /mo > → < eq/ >	0.9993
< mi > n < /mi > → < ci > n < /ci >	0.9941
< mo > - < /mo > → < minus/ >	0.9431
< mo > - < /mo > → < plus/ >	0.0566
< mo > + < /mo > → < plus/ >	0.9995

3.2.4 Content MathML Generation

Segmentation rules and translation rules are applied for the translation at this step. Given a Presentation MathML tree, the system will generate a corresponding Content MathML tree. A greedy translation method is used here to reduce translation time. If more than two rules can be applied to translate a tree, the rule with higher probability is chosen.

The translation process is as follows: First, the same pre-process module is applied on the Presentation MathML tree. The difference here is that the system removes only non-semantic elements. Second, if the processed tree can be translated using translation rules, then apply the rule for translation. If not, the segmentation rule is applied to segment the tree into subtrees. If no rule can be applied, return a translation error. Third, the translation rules are applied to translate the Presentation MathML subtrees into Content MathML subtrees. Finally, the translated Content MathML subtrees are grouped to form the complete Content MathML tree.

Algorithm 2 gives the translation algorithm. The "GetBestRule" function searches for the rule with highest probability in the rule list. The "Apply" function applies a translation rule to a Presentation MathML tree and returns the

translated Content MathML tree. The "RebuildTree" function combines the translated subtrees into a complete tree based on the alignment indexes in the segmentation rule. In some cases, the system was unable to apply any of the segmentation or translation rules, generally due to unseen data. For those cases, the system ignored the root of the subtree and translated its children. This would generate errors at the root of the subtree but improve overall performance. Heuristic translations are also applied to translate numbers and identifiers in the *mn* and *mi* tags.

Algorithm 2 Translate Presentation to Content MathML tree

Input: Presentation MathML tree preTree
segmentation rules SR
translation rules TR
Output: Content MathML tree contentTree
function TRANSLATE($preTree$)
rule1 ← GetBestRule(preTree, TR)
if $rule1 \neq null$ **then**
return Apply(tRule, preTree)
end if
rule2 ← GetBestRule(preTree, SR)
if $rule2 \neq null$ **then**
let pSub[1 .. N] be subtrees of preTree
let cSub[1 .. N] be new contentTree
for $i = 1 \rightarrow N$ **do**
$cSub[i]$ = TRANSLATE((pSub[i]))
end for
return RebuildTree(cSub, sRule)
▷ combines cSub based on the segmentation rule
else
return $< cerror/ >$
end if
end function

Using the proposed approach, the system is capable of handling ambiguous cases. Figure 3.3 shows an disambiguation example. Normally the term $< mi > sin < /mi >$ is translated to <sin/> but when it is accompanied by $< mrow ><$

$mo > - < /mo >< mn > 1 < /mn >< /mrow >$, the system can correctly translated it to $< arcsin/ >$.

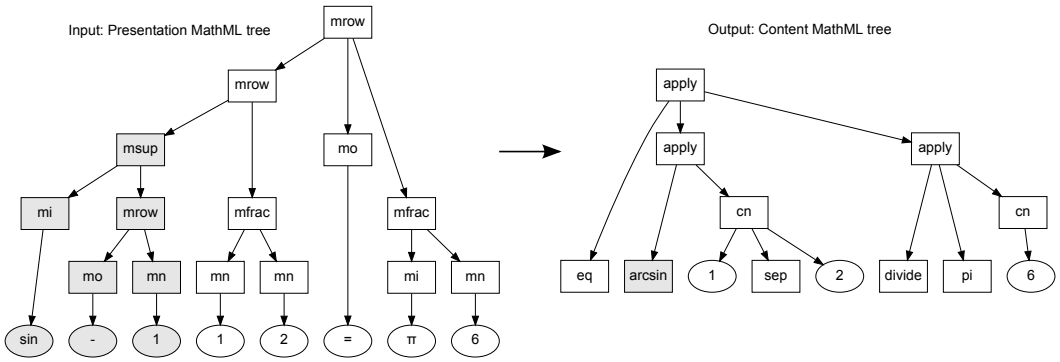

Figure 3.3: Disambiguation example

3.3 Experimental Results and Discussions

3.3.1 Evaluation Setup

The evaluation used two datasets for the experiments:

- The first dataset is Wolfram Functions site data (WFS) and contains mathematical expressions collected from the Wolfram Functions site [Wolfram, 2013], a site created as a resource for educational, mathematical, and scientific communities. The site contains the world's most encyclopedic collection of information on mathematical functions. All formulas on this site are available in both Presentation MathML and Content MathML format. In the experiments, there are six mathematical categories: elementary functions, constants, Bessel-type functions, integer functions, polynomials, and Gamma Beta Erf. The dataset contains 205,653 mathematical expressions in total.

- The second dataset is ACL Anthology Reference Corpus [Bird *et al.*, 2008; Kan, 2013] (ACL-ARC) which contains mathematical expressions extracted from scientific papers in the area of Computational Linguistics and Language Technology. This corpus is also a target corpus of the mathematical formula recognition task in The ACL 2012 Contributed Task [Schafer *et al.*, 2012]. Currently, mathematical expressions are drawn from 20 papers to investigate the cross-domain applicability of the proposed method. All mathematical expressions in these papers are manually annotated with both Presentation Markup and Content Markup. The total number of mathematical expressions in the data set is 2,065. Table 3.3 gives various statistics for these datasets.

The default parameter setting of GIZA++ is used to obtain the alignments between Presentation MathML terms and Content MathML terms.

Table 3.3: Data statistics. The first six categories were collected from the Wolfram Functions site. The last was extracted from 20 ACL papers.

Category	**No. of math expressions**
Bessel-TypeFunctions	1,960
Constants	709
ElementaryFunctions	30,220
GammaBetaErf	2,895
IntegerFunctions	1,612
Polynomials	1,489
ACL-ARC	2,065

3.3.2 Evaluation Methodology

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 is retained as validation data for testing the model, using the

remaining subsets as training data. The cross-validation process was repeated ten times, with each of the ten subsets used exactly once as validation data. The ten results from the folds then averaged to produce a single estimate. In both datasets, formula-wise partition is used.

Given a Presentation MathML expression e, let A is the correct Content MathML tree and B is the output tree of the automatic translation. Evaluation was done by counting the correctness of tree B by comparing it directly to tree A. In the experiments, the system extended the conventional definition of Translation Edit Rate and applied a specific metric that combines the following:

- Tree Edit Distance [Zhang & Shasha, 1989]: The tree edit distance is the minimal cost of transforming A into B using edit operations. Three types of edit operations are possible: substituting, inserting, or deleting a node.
- Translation Edit Rate [Snover *et al.*, 2006]: The translation edit 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.

The new metric is called the Tree Edit Distance Rate (TEDR). TEDR is defined as the ratio of (1) the minimal cost of transforming 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 Equation 4.1.

$$TEDR(A \rightarrow B) = \frac{TED(A, B)}{max\{|A|, |B|\}} \tag{3.3}$$

For example, Figure 3.4 depicts an output tree (A) and a reference (B). Compared to the reference tree, the system must substitute 1 node, insert 3 nodes, and delete 0 node in the output tree, so that $TED(A, B) = 4$, while the maximum number of nodes of the two trees is 8. Therefore, $TEDR(A \rightarrow B) = \frac{4}{8} = 0.5$. $TEDR = 0$ is optimal for this metric.

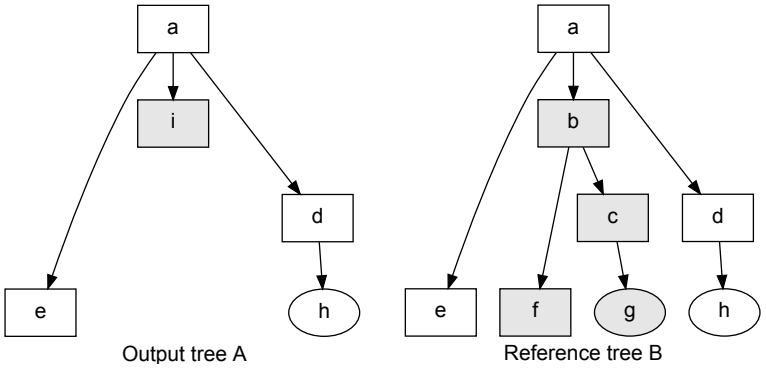

Figure 3.4: Example of an output tree (A) and a reference (B).

There are two methods to calculate the TEDR value of multiple tree pairs. The tree-based method (macro averaged) calculates the TEDR value of each tree pair and then average them. The main disadvantage of this method is that it treats small and large trees equally. The node-based method (micro averaged) first sums up the TED value of these multiple tree pairs and then divides this value to the sum of maximum number of nodes. The second method was chosen because it gives better estimation of how many nodes are wrongly translated. For easier interpretation, the chart representation of the result uses $1 - TEDR$. In this scenario, the higher the $1 - TEDR$ score, the better the system.

Besides TEDR, another metris is used which is the Perfect Translation Rate (PTR). PTR is simply the percentage of perfectly translated expressions. PTR is calculated by counting how many expressions are correctly translated and then divide this number to the number of expressions. This metric is important in applications which require perfect translation, such as searching for exact mathematical expressions.

3.3.3 Experimental Results

First, the evaluation investigated the coverage of segmentation and translation rules which were automatically extracted from the training data. Evaluation was done using the data from the Elementary Functions category, the largest category. Segmentation and translation rules are effective in 98.69% of translation cases. The rest 1.31% is where the system cannot apply any segmentation nor translation rule, which will generate *cerror* node. Translation rules are used twice as often as segmentation rules. Translation rules contribute 65.62% of the translation, while segmentation rules contribute about 33.07%. The accuracy of these rules is 99.13% and 98.3%, respectively. (This value is calculated by the ratio of the correct rules applied to the total rules applied.) The results show that the coverage of segmentation and translation rules is high and selected rules are mostly correct.

Second, the translation quality of the system are then investigated with different mathematical categories. For the WFS dataset, the experimental results showed that the proposed approach gave good results: an 8% TEDR score with a large training data set ("Elementary Functions" category). For smaller data sets (fewer than 3,000 training samples), the results vary from 41% to 49% TEDR. Table 3.4 shows results for each category of the Wolfram Functions Site data. The second and third columns show the average number of segmentation rules and translation rules extracted on each fold, respectively. The two last columns show TEDR and PTR scores.

Third, an experiment is set up using 20,314 short mathematical expressions in the Wolfram Functions Site data. This experiment investigated the correlation between translation quality and the size of the training data set. There are 10 test sets, each test set contain 10 percent of the total mathematical expressions. The training data for each test set varied from 10 to 90 percent of the data size, each

Table 3.4: Results for each category of the Wolfram Functions Site data.

Category	Avg. No. of FR	Avg. No. of TR	TEDR	PTR
Bessel-TypeFunctions	447	9,432	42.31	19.24
Constants	258	1,116	42.35	18.67
ElementaryFunctions	937	12,286	**8.00**	**67.48**
GammaBetaErf	658	8,594	49.30	15.9
IntegerFunctions	431	2,667	41.03	23.2
Polynomials	457	4,464	45.73	13.04

stage added 10 percent of the expressions to the training set. Figure 3.5 shows the correlation between translation quality and training set size. The results of the this experiment are consistent with those of Koehn *et al.* [2003] who found the larger the training data, the better the results. The error rate decreased with the training data size while perfect translation rate increased.

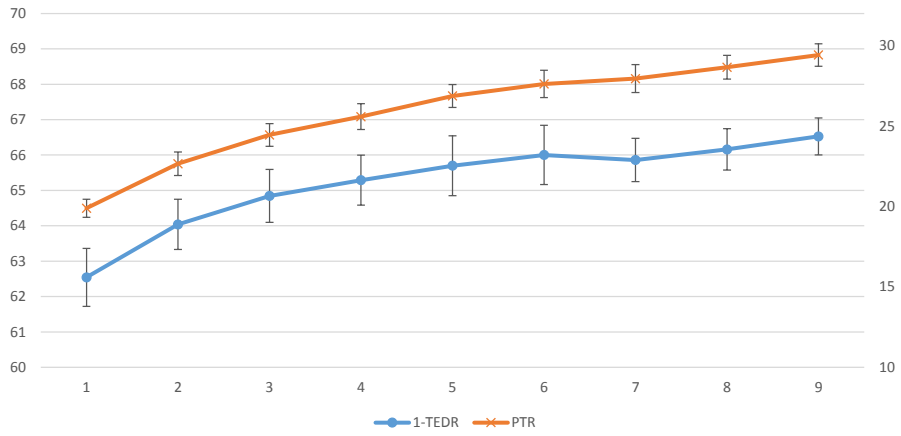

Figure 3.5: Correlation between TEDR and PTR scores and training set size.

Finally, this study set up an experiment to compare the proposed system to SnuggleTeX using ACL-ARC dataset. SnuggleTeX cannot be used with the WFS dataset because the WFS dataset contains a large number of Unicode symbols while SnuggleTeX provides very limited support. There are two systems, SMT-1 used ACL-ARC data for training and testing while SMT-2 used WFS data

for training and ACL-ARC data for testing. Table 3.5 shows the TEDR and PTR scores of the proposed systems compared to SnuggleTeX. SMT-2 system had a 27.67% lower TEDR score and a 4.4% higher PTR score compared to SnuggleTeX. For this cross-domain setting, SMT-based method is advantageous, and even more when the datasets belong to the same domain. SMT-1 system had a 32.69% lower TEDR score and a 16.35% higher PTR score compared to SnuggleTeX, while running times of both systems were more or less equivalent. However, the proposed systems needed to learn the rules from the training data in advance.

Table 3.5: Results for ACL-ARC data. SMT-1 used ACL-ARC data, ten-fold cross-validation. SMT-2 used the rules extracted from WFS Data.

	TEDR	**PTR**
SMT-1	**58.63**	**47.12**
SMT-2	63.65	35.17
SnuggleTeX	91.32	30.77

Figure 3.6 shows the comparison of the different systems. In direct comparison, the systems using SMT are superior to SnuggleTeX in both evaluation metrics. The result of SMT-1 is higher than the result of SMT-2 because this system took advantage of the manually annotated training data of papers from the ACL archive. Surprisingly, all of the systems got better PTR scores compare with PTR scores achieved using WFS data for testing. These results may be explained by the fact that it is easier to correctly translate short mathematical expressions which contain only one or two nodes than long expressions and ACL-ARC data has a large number of short expressions.

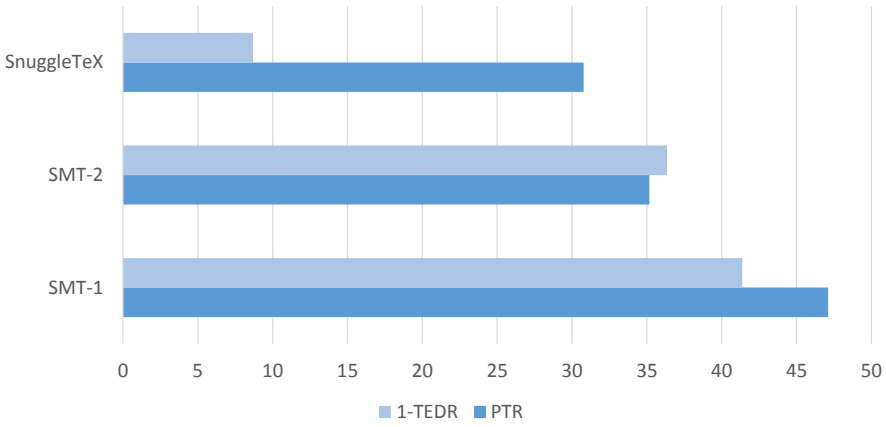

Figure 3.6: Comparison of the different systems.

Chapter 4

Sense Disambiguation of Mathematical Term

This chapter addresses the open problem of mathematical term sense disambiguation. Section 4.1 introduces a method that uses a MathML parallel markup corpus to generate relevant training and testing datasets. Experimental results indicate that we can generate such data automatically and with reasonable accuracy. Based on the dataset generated, in section 4.2, a Support Vector Machines classifier is used to disambiguate the sense of mathematical terms. This approach combines with statistical machine translation approach in chapter 3 improved the *semantic enrichment of mathematical expressions* performance. Portions of this chapter were previously published as [Nghiem *et al.*, 2013b,c].

4.1 Mathematical Sense Disambiguation Data Creation

4.1.1 Overview

Word-sense disambiguation (WSD) refers to the process of identifying the correct sense or meaning of a word in a sentence when the word has multiple meanings. WSD remains a difficult open problem in natural language processing. Current WSD systems are based on supervised, unsupervised, and knowledge-based approaches [Navigli, 2009]. This chapter focuses on the problem of disambiguating the sense of mathematical terms occurring within normal text, an aspect little discussed to date. Mathematical term sense disambiguation will be an enhancement for *semantic enrichment of mathematical expressions* system.

The problem of achieving automated understanding of mathematical expressions can be illustrated quite clearly. For instance, depending on context, the mathematical term δ can be interpreted to refer to `Kronecker Delta`, `Dirac Delta`, `Discrete Delta`, or simply to a variable δ. Another example is `i`, which can be interpreted to mean `the imaginary constant`, `the index variable`, or `the bound variable` of an operation. Other examples include α, β, σ, ϕ, ω, Φ, B, H, x, y, sim. In many such cases, disambiguation can play a crucial role in the automated understanding, translation, and calculation of mathematical expressions.

One major issue in early research on machine understanding of mathematical terms found in text was the lack of evaluation datasets. A previous study [Wolska *et al.*, 2011] was based on a small evaluation set of 200 mathematical expressions annotated by experts. Clearly, large samples of sense-tagged data would require significant human annotation and labor. Fortunately, then, Ide *et al.* [2002] showed that sense distinctions derived from cross-lingual information are at least

as reliable as those made by human annotators. The novel research described here presents a fully automated method for generating large samples of mathematical terms with sense-tagged data.

As part of the effort described here to address mathematical term sense disambiguation (MTSD), this chapter first proposes a method that uses a MathML parallel markup corpus to generate training and testing datasets. Second, this chapter proposes heuristics that improve alignment results for the parallel markup corpus. Third, this chapter presents a classification-based approach to the MTSD problem. To the best of our knowledge, this study is the first to make use of parallel corpora to address MTSD.

4.1.2 Method

This method compiled the MTSD data using parallel MathML markup expressions gathered from the Web. First, using a set of heuristic rules, the system preprocessed the parallel MathML markup expressions. It then used the GIZA++ toolkit to obtain node-to-node aligned data. Based on the node-to-node aligned data, the system created subtree-to-subtree aligned data. Finally, it extracted ambiguous terms from the subtree-to-subtree aligned data to obtain data for MTSD. Figure 4.1 gives the steps taken to generate the data.

A crucial step in generating MTSD data is achieving alignment between the Presentation side and the Content side of the expressions. Given a set of several MathML parallel markup expressions, the system used the automated word alignment GIZA++ [Och & Ney, 2003] to obtain alignment between the Presentation terms and Content terms. Developed to train word-based translation models, the GIZA++ toolkit is not directly applicable to a tree-based corpus. One common solution is to convert the tree into a sentence by extracting the leaf nodes of the tree and to form a sequence [Sun *et al.*, 2010]. While this approach works well

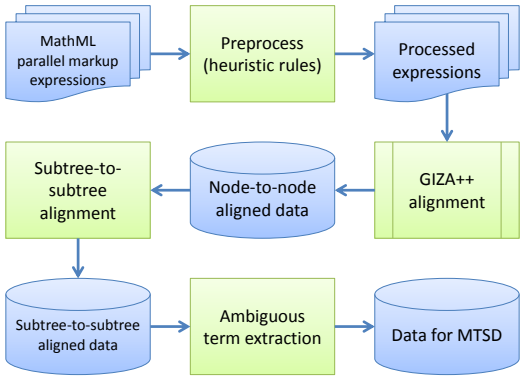

Figure 4.1: Steps for generating the data for MTSD.

for natural language text, it is less effective with mathematical expressions, since the intermediate nodes of these expressions contain layout information.

Before using GIZA++, to enhance alignment precision, the system applies two heuristic rules to the presentation tree based on information on its structure. The first heuristic rule converts the intermediate layout nodes (except `mrow`) to leaves on the tree by moving them to the position of their first child. When moving an intermediate layout node, the system creates a temporary ('temp') node to replace the moved node and to keep the other child nodes intact. Unnecessary parentheses, which indicate that the expressions in the parentheses belong together, are also removed. Figure 4.2 illustrates an example of this heuristic.

In Figure 4.2, the system moved the `msup` node to a leaf of the tree and removed a pair of parentheses, `<mo>(</mo>` and `<mo>)</mo>`, near `<mn>0</mn>` node. Red lines represent alignments from presentation nodes to content nodes. Green lines represent alignments from content nodes to presentation nodes. Blue lines represent expanded alignments between subtrees.

The second heuristic rule moves operator (`mo`) nodes to the beginning of the subtree if that subtree contains operator nodes. This rule reduces cross alignments, since most notations in content MathML are prefix notations and placed in leaf nodes. In Figure 4.2, the `<mo>=</mo>` node is moved to the first position

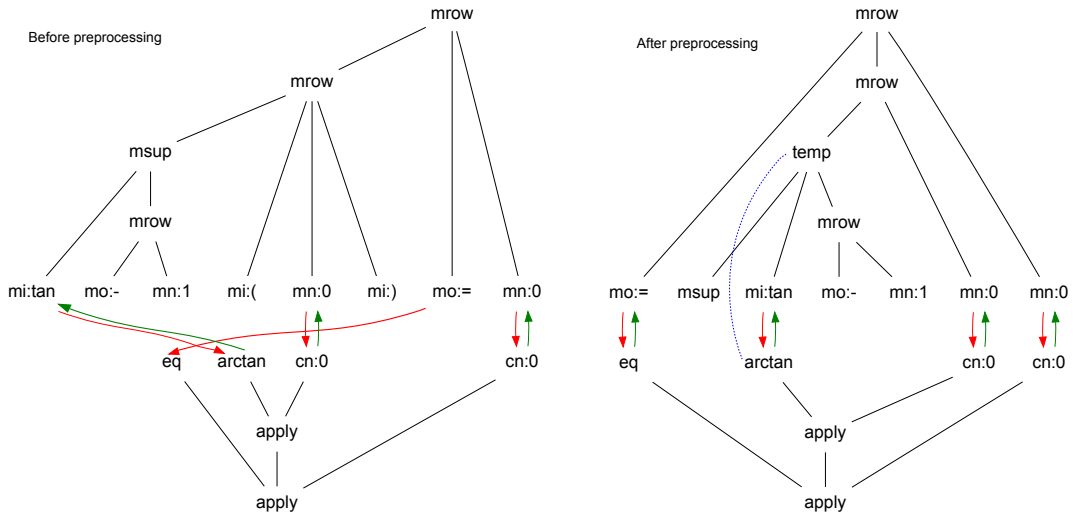

Figure 4.2: Example of alignment results for GIZA++ before and after applying the heuristic rules for the expression `arctan(0)=0`.

of the tree. The `<mo>-</mo>` node is not moved because it is already the first child of its parent node.

To extract more complex mathematical terms, the system expand the node-to-node alignments to subtree-to-subtree alignments. In this study, the subtree alignment is expanded only to the parent of the `mi` nodes. The criteria used here to achieve subtree aligned pair are similar to that used by Tinsley *et al.* [2007]. First, a node can be linked only once. Second, descendants of a presentation node can link only to descendants of its content counterpart. Third, ancestors of a presentation node can link only to ancestors of its content counterpart (a node counts as its own ancestor).

If one presentation node links to more than one content node, the system keeps only the link with the highest alignment score, as given by Equation 4.1. The number of alignments between the presentation tree $tree_P$ and the content tree $tree_C$ is the sum of (1) the number of alignments from the leaf children of $tree_P$ to the leaf children of $tree_C$ and (2) the number of alignments from the leaf children of tree to the leaf children of $tree_P$. For more accurate results, the

system removed node-to-node alignments if alignment probabilities fell below a certain threshold (0.2). In Equation 4.1, P_{child} and C_{child}, respectively, refer to the child nodes of $tree_P$ and $tree_C$. The blue lines in Figure 4.2 represent the expanded alignments between subtrees.

$$score(\text{tree}_\text{P}, \text{tree}_\text{C}) = \frac{\#\text{ alignments}}{\#\text{ P}_\text{child} + \#\text{ C}_\text{child}} \tag{4.1}$$

Based on the alignment results, the system extracted pairs of presentation mathematical terms and their associated content terms. A mutually aligned presentation subtree and content subtree form a pair. This study will consider only mathematical terms containing `mi` (e.g. tan^{-1}, `Ai`, `Ai(0)`, Γ, $\Gamma(\frac{2}{3})$). Only terms associated with ambiguous mapping are retained to generate training and testing data.

4.2 Mathematical Sense Disambiguation System

4.2.1 Overview

Disambiguation of mathematical elements is an important component in the semantic enrichment system. Basic methods for dealing with ambiguities so far were rule-based [McKain, 2013; Miller, 2013]. The rule-based approach is of course generally not able to derive meaning from arbitrary Presentation MathML expressions. The statistics-based approach resolves ambiguities based on the probabilities Nghiem *et al.* [2013a], and thus gets better results than the rule-based system. This chapter enhances the statistics-based approach by combining it with a disambiguation component.

So far, there has been limited discussion about the contribution of surrounding text to mathematical element disambiguation problem. It is becoming increas-

ingly difficult to ignore the surrounding text of mathematical expressions. For example, the token δ can be mapped to $KroneckerDelta$ if its surrounding text contains the word 'Kronecker delta'. It is difficult to disambiguate using only the presentation of mathematical expression. The combination of mathematical expression itself and its surrounding text can lead to improvements in disambiguation process.

The aim of this section is to examine and solve the ambiguity when mapping Presentation MathML elements to their Content elements. This section also attempts to find the contribution of surrounding text to mathematical element disambiguation problem. A Support Vector Machine [Cortes & Vapnik, 1995] (SVM) learning model is used for MathML Presentation token element (mi) disambiguation. Both presentation of mathematical expression and its surrounding text are encoded in a feature vector used in SVM. To evaluate the efficacy of the proposed method, the system is incorporated into an SMT-based semantic enrichment system.

The problem is formulated as follows: given a Presentation MathML expression and its surrounding text, can we interpret its Content MathML expression? This chapter provides contributions in three main areas of mathematical semantic enrichment problem. First, it shows that combination of a disambiguation component and the SMT-based system improves the system's performance. Second, it shows that the text surrounding the mathematical expressions contributes to the disambiguation process. Third, it shows that the name of the category that a mathematical expression belongs to is the most important text feature for disambiguation.

4.2.2 Method

The system has two phases, a training phase and a running phase, and consists of three main modules.

- Statistical-based rule extraction: Extracts rules for translation, given the training data. Two types of rules are established: segmentation rules and translation rules. Each rule is associated with its probability.
- SVM-based disambiguation: An SVM training algorithm builds a model that assigns to identifiers (mi) their correct content. Features are extracted from both the presentation of mathematical expressions and their surrounding text.
- Translation: The input of this module includes a Presentation MathML expression, a set of rules for translation, and the output from the disambiguation module. This module translates Presentation into Content MathML expression.

Figure 4.3 shows the system framework.

4.2.2.1 Statistical-based rule extraction

The rules for translation were extracted according to the procedure in chapter 3. Given a set of training mathematical expressions in MathML parallel markup, two types of rules are extracted: segmentation rules and translation rules. Translation rules are used to translate (sub)trees of Presentation MathML markup to (sub)trees of Content MathML markup. Segmentation rules are used to combine and reorder the (sub)trees to form a complete tree. The output of this module is a set of segmentation and translation rules, each rule is associated with its probability.

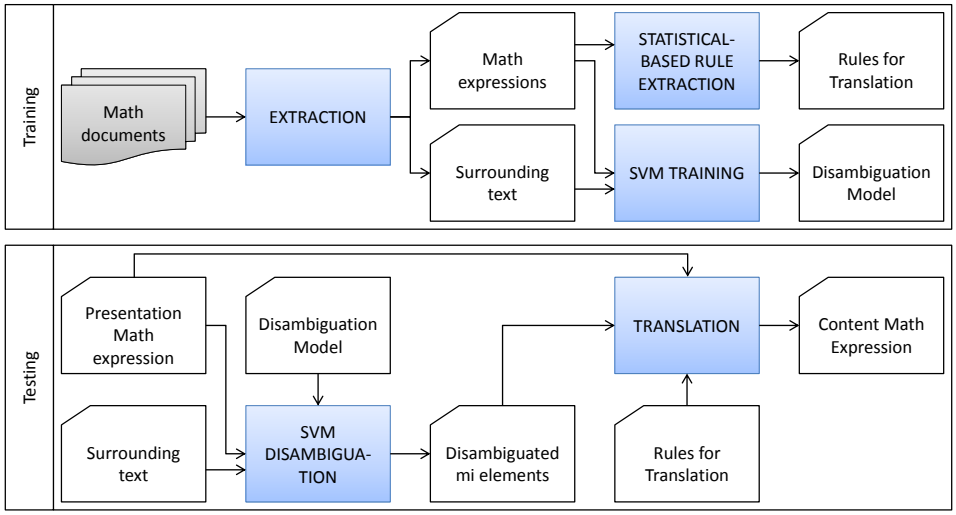

Figure 4.3: System Framework

4.2.2.2 SVM disambiguation

An *mi* token element in MathML presentation markup can be translated into many different elements in MathML content markup. In this section, it is assumed that one *mi* element can be translated into one of a limited predefined set of Content elements. Given an *mi* element, the system uses an SVM training algorithm to build a model that assigns to its correct Content element. When translating, each of the Presentation *mi* elements will be disambiguated before generating Content MathML expressions. The accuracy of the SVM disambiguation is a crucial preprocessing step for a high-quality MathML Presentation to Content translation.

The system used the alignment output of GIZA++[1] [Och & Ney, 2003] to generate training and testing data for the disambiguation problem. Given a training data consists of several parallel markup expressions, GIZA++ was used to align the Presentation terms to the Content terms. From this alignment results, the system extracts pairs of Presentation *mi* elements and their associated Content

[1]https://code.google.com/p/giza-pp/

elements. Only *mi* elements that have ambiguities in their translation are kept to generate training and testing data. Table shows 4.1 the examples of Presentation *mi* elements and their associated Content elements.

Table 4.1: Presentation *mi* elements and their associated Content elements

Presentation elements	**Content elements**
<mi> σ </mi>	<ci>Weierstrass Sigma</ci>
	<ci>Divisor Sigma</ci>
	<ci> σ </ci>
<mi> μ </mi>	<ci>MoebiusMu</ci>
	<ci> μ </ci>
<mi>H</mi>	<ci>StruveH</ci>
	<ci>Harmonic Number</ci>
	<ci>Hankel H1</ci>
	<ci>Hankel H2</ci>
	<ci>Hermite H2</ci>
	<ci>H</ci>
<mi>y</mi>	<ci>Bessel Y Zero</ci>
	<ci>Spherical Bessel Y</ci>
	<ci>y</ci>

For each mathematical expression, an *mi* element has only one correct translation. In other mathematical expressions, the same *mi* element might have another correct translation. Assume that an *mi* element e has n ways of translating from Presentation into Content MathML. For each mathematical expression, the system creates one positive instance by combining e and its correct translation. The system also creates $n - 1$ negative instances by combining e and its incorrect translations.

The features used in the SVM disambiguation may be divided into two main groups: Presentation MathML features and surrounding text features. Presentation MathML features are extracted from the Presentation MathML markup of

the mathematical expression. Surrounding text features are extracted from the text surrounding the mathematical expression. The category which the mathematical expression belongs to is also used. Table 4.2 shows the features used for classification.

Table 4.2: Features used for classification

Feature		**Description**
Presentation MathML feature	Only child	Is it the only child of its parent node
	Preceded by mo	Is it preceded by an <mo> node
	Followed by mo	Is it followed by an <mo> node
	⁡	Is it followed by a Function Application
	Parent's name	The name of its parent node
	Name	The name of the identifier
Text feature	Category	Relation between category name and candidate translation
	Unigram	Vector represents unigram feature
	Bigram	Vector represents bigram feature
	Trigram	Vector represents trigram feature
Candidate translation		One of n candidate translations of the *mi* element

There were six Presentation MathML features in this experiment. The first one determines whether the *mi* element is the only child of its parent. The relation between the *mi* element and its surrounding *mo* elements is encoded in the following three features. The last two features represent the name of the *mi* element and its parent. Among these features, the name of the *mi* element is the

most important feature.

Among the text features, the first one is the category that mathematical expression belongs to. In mathematical resource websites, such as the Wolfram Functions Site, mathematical expressions belong to different categories. However we usually do not have the text surrounding these mathematical expressions. The system then can calculate the relation between the category name and the Content translation of each *mi* element. The relation has one of three values: the same as the Content translation, contains the Content translation, or does not contain the Content translation.

In case there are available the text surrounding or the description of the mathematical expressions, the system can use n-gram features [Cavnar & Trenkle, 1994]. The system uses unigram, bigram and trigram features in this study. These features are implemented as the vectors containing the n-grams which appear in the training data. The system will assign each instance into one of two classes, depending on the candidate translation. The class is 'true' if the candidate translation is the correct Content translation of the *mi* element, and 'false' otherwise.

Each training instance of SVM learning is a vector which contains Presentation MathML features, text features, guessed meaning, and a Boolean variable indicates whether the guessed class is correct or not. The number of text features are depending on the dataset. For the Wolfram Functions Site data, each training instance contains one category feature. For the ACL data, each training instance contains three n-gram features: unigram, bigram and trigram features. When running, the system generates some meanings for each mi term and SVM will decide which meaning is the correct meaning. Since the binary decision is made independently for each MathML term, SVM might decide that there are two or more correct meanings for one term. In such a case, the system choose

the meaning which has higher probability.

4.2.2.3 Translation

After disambiguation, the result is used to enhance the semantic enrichment of a statistical-machine-translation-based system. The input of this module includes a Presentation MathML expression, a set of rules for translation, and the output from the disambiguation module. The output of this module is the Content MathML expression which represents the meaning of the Presentation MathML expression. If there is only one mapping from a Presentation element, that Content element is chosen. If the disambiguation module accepts more than two mappings from a Presentation element, the Content element with higher probability is chosen.

4.3 Evaluation

4.3.1 Mathematical Sense Disambiguation Data Creation

4.3.1.1 Evaluation Setup

For these experiments, the data was collected by using parallel MathML markup expressions from the Wolfram Functions Site as described in 3. All mathematical expressions on WFS are available in MathML parallel markup. For simplicity, the system excluded long expressions containing more than 30 leaf nodes. After that, there is a total of 20,314 mathematical expressions on the dataset.

4.3.1.2 Evaluation Results

Evaluation began by investigating the quality of the generated MTSD data. Using WFS data, the system generated 2,925 different mathematical terms. There are

390 distinct ambiguous terms and 2,535 distinct unambiguous terms. Of the ambiguous terms, 90 distinct terms are single `mi` elements. There are 67,987 instances contain all the ambiguous terms in the data. Table 4.3 shows the generated data.

Table 4.3: Generated data

Type	**Distinct term**
Ambiguous `mi` terms	90
Other ambiguous terms	300
Unambiguous terms	2,535

The table shows that only 14% of the extracted mathematical terms are ambiguous. One possible explanation: in WFS data, people tend to use one meaning for a fixed notation. Another: the system depends on the quality of the alignment output. The aligner may ignore an alignment if the probability of the alignment is low. This also causes errors in sense extraction if a sub-tree is aligned with a single term but the links are not fully connected: for example, tan^{-1} (Presentation) and *arctan* (Content).

Within the scope of this study, the system focused on the single mi element terms. They was chosen because of the mi element term often contains more ambiguities than other terms: number (mn) and operator (mo) terms. The same method can be expanded to encompass additional ambiguous terms. These single mi element terms are manually verified to assess the quality of the generated MTSD data. Of 247 extracted senses, 197 were correct, an accuracy rate of 79.76% for the generated data. Each mi element term has an average of 2.74 senses. The term with the most senses was `<mi>C</mi>`, which had six senses: Catalan, CatalanNumber, C, GegenbauerC, Cyclotomic, and FresnelC.

Next, evaluation continues by setting up an experiment using libSVM[1] in the Weka toolkit [Hall *et al.*, 2009] to examine sense disambiguation results for each presentation MathML term. The data which was used contained the 90 distinct ambiguous mi terms. Evaluation compared the results for systems using different training data: automatically extracted data and manually verified data. The system also compared the results of this approach to the 'most frequent' method, which chooses the interpretation of highest probability. Since in the real world not every mathematical expression is associated with its category name, another experiment is also set up to assess the performance of this approach with and without the 'category' feature.

The system built two models using nine-tenths of the automatically extracted data and nine-tenths of the manually verified data. Both systems set aside one-tenth of the verified data for testing. Classification accuracies were computed over the set of binary decisions. The default libSVM parameters are used. Table 4.6 gives the disambiguation accuracy for ambiguous `mi` terms.

Table 4.4: Sense disambiguation accuracy for ambiguous `mi` terms

Method	**Extracted data**	**Verified data**
All feature	**91.40**	**93.94**
Without 'category' feature	91.22	92.41
Most frequent	85.01	89.76

The results in Table 4.6 indicate reasonable results for the automatically extracted data. The proposed approach gained improvements ranging from 1.2 to 2.5 percent by building a model using manually verified data. The classifier with 'category' feature slightly outperformed the classifier without the 'category' fea-

[1] http://www.csie.ntu.edu.tw/∼cjlin/libsvm/

ture. Overall, the results here were approximately 4 to 7 percent more accurate than for the 'most frequent' method. The explanation for the relatively high scores for the 'most frequent' method is that mathematical elements often have a preferred meaning.

The results suggest we can make direct use of automatically generated data when working on the MTSD problem. For mathematical expressions in MathML parallel markup, the generated data is good enough without manual checking. The results also show that the text feature-i.e., the category of the mathematical term-contributes to system performance. While this improvement is modest, it suggests that features aside from the mathematical term itself can be helpful. However, the system works well even without this feature.

4.3.2 Mathematical Sense Disambiguation System

The first dataset for the experiments is the Wolfram Functions site [Wolfram, 2013]. This site was created as a resource for educational, mathematical, and scientific communities. All formulas on this site are available in both Presentation MathML and Content MathML format. The only text information on this dataset is the function category of each mathematical expression. The experiments used 136,685 mathematical expressions divided into seven categories: elementary functions, constants, Bessel-type functions, integer functions, polynomials, Gamma Beta Erf, and polynomials.

The second dataset for the experiments is the Archives of the Association for Computational Linguistics Corpus [Kan, 2013] (ACL-ARC). It contains mathematical expressions extracted from scientific papers in the area of Computational Linguistics and Language Technology. Currently, evaluation use mathematical expressions drawn from 20 papers which were selected from this dataset. All mathematical expressions are manually annotated with MathML parallel Markup

and their textual descriptions. Out of 2,065 mathematical expressions in the dataset, only 648 expressions have their own description. Table 4.5 shows examples of mathematical expressions and their description in ACL-ARC dataset.

The annotation design for linking mathematical formulas to natural language descriptions in the surrounding text are reported in [Kristianto *et al.*, 2012]. There are two types of description: a short description and a full description. Short description specifies the type and category of the mathematical expressions. While full description contains the characteristics of the formula within the category. In our experiment, the full descriptions are used since they contains more information for disambiguation.

The evaluation was done using two metrics: accuracy score for disambiguation and tree edit distance rate score for semantic enrichment. The accuracy score of disambiguation is the ratio of correctly classified instances to total instances. The tree edit distance rate (TEDR) score [Snover *et al.*, 2006] is defined as the ratio of (1) the minimal cost of transforming the generated into the reference Content MathML tree using edit operations and (2) the maximum number of nodes of the generated and the reference Content MathML tree. Evaluation also compares the semantic enrichment results to the results of the system in chapter 3.

First evaluation set up an experiment to examine the disambiguation result on each Presentation MathML *mi* element. In this experiment, three systems are compared. The first system uses both Presentation MathML and text features. The second system uses only Presentation MathML features. The last system chooses the interpretation with highest probability. Table 4.6 shows the results of the disambiguation component.

The results in Table 4.6 show that disambiguation result using SVM outperformed the 'most frequent' method. The reason 'most frequent' method got high scores is because mathematical elements often have a preferred meaning.

Table 4.5: Examples of mathematical expressions and their description in ACL-ARC dataset

Textual description	**MathML Presentation expression**	**MathML Content expressions**
a word to be translated	<mrow> <mi>w</mi> </mrow>	<ci>w</ci>
a word in a dependency relationship	<mrow> <mi>w</mi> </mrow>	<ci>w</ci>
a matrix	<mrow> <mi>t</mi> </mrow>	<ci>t</ci>
a similarity matrix which specifies the similarity between individual elements	<mrow> <mi>sim</mi> </mrow>	<ci>sim</ci>
argument	<mrow> <msub> <mi>S</mi> <msub> <mi>j</mi> <mi>i</mi> </msub> </msub> </mrow>	<apply> <selector /> <ci>S</ci> <apply> <selector /> <ci>j</ci> <ci>i</ci> </apply> </apply>
The LM probabilities	<mrow> <mi>P</mi> <mo>e</mo> <mrow> <mo>(</mo> <mrow> <mi>v</mi> <mo> | </mo> <mrow> <mi>Parent</mi> <mo>e</mo> <mrow> <mo>(</mo> <mi>v</mi> <mo>)</mo> </mrow> </mrow> </mrow> <mo>)</mo> </mrow> </mrow>	<apply> <ci>P</ci> <apply> <ci> | </ci> <ci>v</ci> <apply> <ci>Parent</ci> <ci>v</ci> </apply> </apply> </apply>

Table 4.6: Disambiguation accuracy

Category	Number of instances	With text features	Without text features	Most frequent
ACL-ARC	2,996	92.9573	**93.7583**	93.4246
BesselTypeFunctions	1,352	**92.8254**	92.3077	86.0947
Constants	714	**91.1765**	90.3361	83.7535
ElementaryFunctions	6,073	96.1963	**96.3774**	89.6427
GammaBetaErf	3,816	**95.2830**	94.4706	78.0136
HypergeometricFunctions	72,006	**97.5571**	97.0697	88.0746
IntegerFunctions	11,955	**95.8009**	95.1652	90.0711
Polynomials	5,905	**98.2388**	95.3091	87.3328
All WFS Data	320,726	**98.9243**	98.4398	92.7025

The systems that used only Presentation MathML features achieved even better scores, because they use surrounding mathematical elements. It is interesting to note that on the ACL-ARC data, the 'most frequent' system get higher score than the system with text features. Overall, on WFS data, the system gained 5 to 16 percent accuracy improvements.

The systems that also used text features outperform the systems that used only Presentation MathML features in most of WFS categories. This result may be explained by the fact that the category of a mathematical expression is closely related to that expression. Contrary to expectations, this study did not find any improvement in ACL-ARC data. It seems possible that these results are due to the lack of training data and the sparseness of n-gram features. This finding was unexpected and suggests that in order to use n-gram text features, more data is needed.

Second, evaluation set up an experiment to examine the semantic enrichment

Table 4.7: Semantic enrichment TEDR

Category	Number of expression	With text feature	Without text feature	Most frequent
Bessel-TypeFunctions	701	**18.0604**	**18.0604**	18.4118
Constants	555	**33.9016**	34.0328	34.6230
ElementaryFunctions	9,537	7.4879	**7.4809**	7.7343
GammaBetaErf	1,558	**17.2308**	17.2851	18.4796
Hypergeometric-Functions	9,347	**49.4678**	49.4797	49.6902
IntegerFunctions	1,175	**20.5292**	20.5874	20.9945
Polynomials	727	**19.6309**	19.7987	20.2685
All WFS Data	23,600	**29.0707**	29.0869	29.2769

result. The results from disambiguation component are used in the semantic enrichment system. This evaluation compares three systems: with text feature, without text feature, and the proposed system in chapter 3 which used 'most frequent' method. This experiment uses 90 percent of expressions for training both SVM-based disambiguation and translation components. The evaluation uses the other 10 percent of expressions for testing. Table 4.7 shows the translation result.

The results in Table 4.7 show that combining disambiguation and statistical machine translation improved the system. Expressions in 'Gamma Beta Erf' category benefit from the disambiguation module the most with 1.2 percent error rate reduction. Less ambiguity in elementary functions might lead to lower performance in 'Elementary Functions' category. This part did not show the evaluation result on ACL-ARC data because the disambiguation result was almost the same as the 'most frequent' method. Overall, on WFS data, the proposed approach achieved 0.2 to 1.2 percent error rate reduction.

Chapter 5

Content-based mathematical search

This chapter presents a description of a method for *content-based mathematical search system* and the contribution of *semantic enrichment of mathematical expressions* to that system.

5.1 Overview

The issue of retrieving mathematical content has received considerable critical attention Aizawa *et al.* [2013]. Mathematical content is a valuable information source for many users and is increasingly available on the Web. Retrieving this content is becoming more and more important.

Conventional search engines, however, do not provide a direct search mechanism for mathematical expressions. Although these search engines are useful to search for mathematical content, these search engines treat mathematical expressions as keywords and fail to recognize the special mathematical symbols and constructs. As such, mathematical content retrieval remains an open issue.

Some recent studies have proposed mathematical retrieval systems based on the structural similarity of mathematical expressions Altamimi & Youssef [2008]; Miner & Munavalli [2007]; National Institute of Standards and Technology [2013]; Springer [2013]; Youssef [2005]; Youssef & Altamimi [2007]. However, in these studies, the semantics of mathematical expressions is still not considered. Because mathematical expressions follow highly abstract and also rewritable representations, structural similarity alone is insufficient as a metric for semantic similarity.

Other studies Adeel *et al.* [2008]; Kohlhase & Prodescu [2013]; Kohlhase & Sucan [2006]; Nguyen *et al.* [2012]; Wolfram [2013]; Yokoi & Aizawa [2009] have addressed semantic similarity of mathematical formulae, but this required content-based mathematical formats such as content MathML Ausbrooks *et al.* [2010] and OpenMath Buswell *et al.* [2004]. Because almost all mathematical content available on the Web is presentation-based, these studies used two freely available toolkits, SnuggleTeX McKain [2013] and LaTeXML Miller [2013], for semantic enrichment of mathematical expressions. However, much uncertainty remains about the relation between the performance of mathematical search system and the performance of the semantic enrichment component.

Based on the observation that mathematical expressions have meanings hidden in their representation, the primary goal of this chapter is making use of mathematical expressions' semantics for mathematical search. To accomplish this problem of retrieving semantically similar mathematical expressions, we use the results of state-of-the-art semantic enrichment methods. This chapter seeks the answers to two questions.

- What is the contribution of semantic enrichment of mathematical expressions to content-based mathematical search systems?
- Which one is better: presentation-based or content-based mathematical

search?

To implement a *mathematical search system*, various challenges must be overcome. First, in contrast to text which is linear, mathematical expressions are hierarchical: operators have different priorities, and expressions can be nested. The similarity between two mathematical expressions is decided first by their structure and then by the symbols they contain Kamali & Tompa [2009, 2013]. Therefore, current text retrieval techniques cannot be applied to mathematical expressions because they only consider whether an object includes certain words. Second, mathematical expressions have their own meanings. These meanings can be encoded using special markup languages such as Content MathML or OpenMath. A few existing mathematical search systems also make use of this information. Such markup, however, is rarely used to publish mathematical knowledge related to the Web Kamali & Tompa [2009]. As a result, we were only able to use presentation-based markup, such as Presentation MathML or TEX, for mathematical expressions.

This chapter presents an approach to a *content-based mathematical search system* that uses the information from *semantic enrichment of mathematical expressions* system. To address the challenges described above, the proposed approach is described below. First, the approach used Presentation MathML markup, a widely used markup for mathematical expressions. This makes our approach more likely to be applicable in practice. Second, a *semantic enrichment of mathematical expressions* system is used to convert mathematical expressions to Content MathML. By getting the underlying semantic meanings of mathematical expressions, a *mathematical search system* is expected to yield better results.

5.2 Methods

The framework of the system is shown in Fig. 5.1. First, the system collects mathematical expressions from the web. Then the mathematical expressions are converted to Content MathML using a *semantic enrichment of mathematical expressions* system described in Chapter 3. Indexing and ranking the mathematical expressions are done using Apache Solr system [Apache, 2013] following the method described in Topic *et al.* [2013]. When a user submits a query, the system also converts the query to Content MathML. Then the system returns a ranked list of mathematical expressions corresponding to the user's queries.

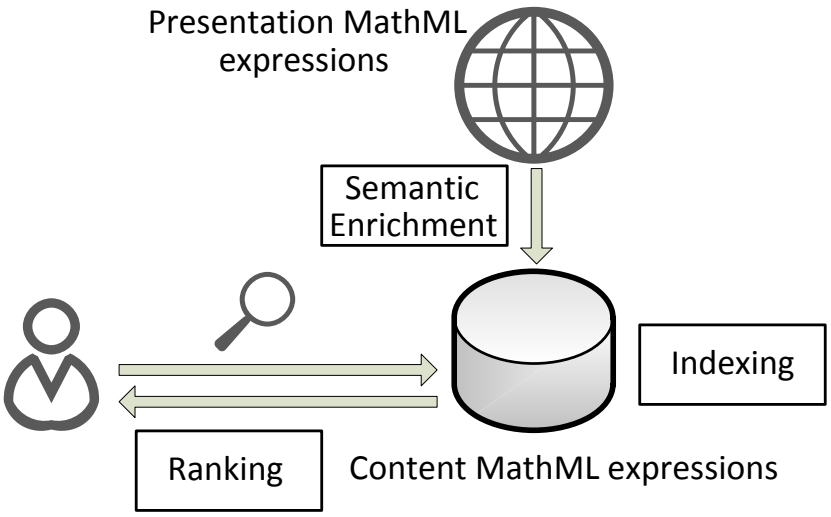

Figure 5.1: System Framework.

5.2.1 Data collection

Performance analysis of a mathematical search system is not an easy task because few standard benchmark datasets exist, unlike other more common information retrieval tasks. Mathematical search systems normally build their own mathematical search dataset for evaluation by crawling and downloading mathematical

content from the web. Direct comparison of the proposed approach with other systems is also hard because they are either unavailable or inaccessible.

Recently, simpler and more rapid tests of mathematical search system have been developed. The NTCIR-10 Math Pilot Task Aizawa *et al.* [2013] was the initial attempt to develop a common workbench for mathematical expressions search. Currently, the NTCIR-10 dataset contains 100,000 papers and 35,000,000 mathematical expressions from ArXiv Cornell University Library [2013] which includes Content MathML markup. The task was completed as an initial pilot task showing a clear interest in the mathematical search. However, the Content MathML markup expressions are generated automatically using the LaTeXML toolkits. Therefore, this dataset is unsuitable to serve as the gold standard for the research described in the present chapter.

As Wolfram Functions Site Wolfram [2013] is the only website that provides high-quality Content MathML markup for every expression, data for the search system was collected from this site. The Wolfram Functions Site data have numerous attractive features, including both Presentation and Content MathML markups, and category for each mathematical expression. In the experiment, the performance of *semantic enrichment of mathematical expressions* component will be compared directly with the system performance obtained using correct Content MathML expressions on Wolfram Functions Site data.

5.2.2 Semantic enrichment of mathematical expressions

The mathematical expressions were preprocessed according to the procedure described in Chapter 3. Given a set of training mathematical expressions in MathML parallel markup, rules of two types are extracted: segmentation rules and translation rules. These rules are then used to convert mathematical expressions from their presentation to their content form. Translation rules are used

to translate (sub)trees of Presentation MathML markup to (sub)trees of Content MathML markup. Segmentation rules are used to combine and reorder the (sub)trees to form a complete tree.

After using the semantic enrichment of mathematical expressions system to convert the expressions into content MathML, we use these converted expressions for indexing. The conversion is not a perfect conversion, so there are terms that could not be converted. The queries submitted to the search system are also processed using the same conversion procedure.

5.2.3 Indexing

The indexing step was prepared by adapting the procedure used by Topić et. al Topic *et al.* [2013]. This procedure used *pq*-gram-like indexing for Presentation MathML expressions. We modified it for use with Content MathML expressions. There are three fields used to encode the structure and contents of a mathematical expression: `opaths`, `upaths`, and `sisters`. Each expression is transformed into a sequence of keywords across several fields. `opaths` (ordered paths) field gathers the XML expression tree in vertical paths with preserved ordering. `upaths` (unordered paths) works the same as `opaths` without the ordering information. `sisters` lists the sister nodes in each subtree. Figure 5.2 presents an example of the terms used in the index of the expression $\sin(\frac{\pi}{8})$:$< apply >< sin/ >< apply >< times/ >< pi/ >< apply >< power/ >< cntype = \text{“}integer\text{”} > 8 < /cn >< cntype = \text{“}integer\text{”} > -1 < /cn >< /apply >< /apply >< /apply >$.

5.2.4 Searching

In the mathematical search system, users can input mathematical expressions using presentation MathML as a query. The search system then uses the *semantic enrichment of mathematical expressions* module to convert the input expressions

opaths:
`1#1#apply 1#1#1#sin 1#1#2#apply 1#1#2#1#times 1#1#2#2#pi 1#1#2#3#apply 1#1#2#3#1#power 1#1#2#3#2#cn#8 1#1#2#3#3#cn#-1`

opaths:
`1#apply 1#1#sin 1#2#apply 1#2#1#times 1#2#2#pi 1#2#3#apply 1#2#3#1#power 1#2#3#2#cn#8 1#2#3#3#cn#-1`

opaths:
`apply 1#sin 2#apply 2#1#times 2#2#pi 2#3#apply 2#3#1#power 2#3#2#cn#8 2#3#3#cn#-1`

opaths: `sin`

opaths: `times`

opaths: `pi`

opaths: `apply 1#power 2#cn#8 3#cn#-1`

opaths: `power`

opaths: `cn#8`

opaths: `cn#-1`

upaths:
`##apply ###sin ###apply ####times ####pi ####apply #####power #####cn#8 #####cn#-1`

upaths:
`#apply ##sin ##apply ###times ###pi ###apply ####power ####cn#8 ####cn#-1`

upaths:
`apply #sin #apply ##times ##pi ##apply ###power ###cn#8 ###cn#-1`

upaths: `sin`

upaths: `apply #times #pi #apply ##power ##cn#8 ##cn#-1`

upaths: `times`

upaths: `pi`

upaths: `apply #power #cn#8 #cn#-1`

upaths: `power`

upaths: `cn#8`

upaths: `cn#-1`

sisters: `power cn#8 cn#-1`

sisters: `times pi apply`

sisters: `sin apply`

sisters: `apply`

Figure 5.2: Index terms of the expression $\sin(\frac{\pi}{8})$.

opaths:
`1#1#apply 1#1#1#sin 1#1#2#apply 1#1#2#1#times 1#1#2#2#pi 1#1#2#3#apply 1#1#2#3#1#power 1#1#2#3#2#cn#8 1#1#2#3#3#cn#-1`
upaths:
`##apply ###sin ###apply ####times ####pi ####apply #####power #####cn#8 #####cn#-1`
upaths:
`#apply ##sin ##apply ###times ###pi ###apply ####power ####cn#8 ####cn#-1`
sisters: `power cn#8 cn#-1`
sisters: `times pi apply`
sisters: `sin apply`
sisters: `apply`

Figure 5.3: Query terms of the expression $\sin(\frac{\pi}{8})$.

to Content MathML. Figure 5.3 presents an example of the terms used in the query of the expression $\sin(\frac{\pi}{8})$. Matching is then performed using eDisMax, the default query parser of Apache Solr. Ranking is also done using the default modified TF/IDF scores and length normalization of Apache Solr.

5.3 Experimental Results

5.3.1 Evaluation Setup

We collected mathematical expressions for evaluation from the Wolfram Function Site. At the time collected, there were more than 300,000 mathematical expressions on this site. After collection, we filtered out long expressions containing more than 20 leaf nodes to speed up the semantic enrichment because the processing time increases exponentially with the length of the expressions. The number of mathematical expressions after filtering is approximately 20,000. Presumably, this number is adequate for evaluating the mathematical search system.

Evaluation was done by comparing three systems:

- Presentation-based search with Presentation MathML (PMathML): indexing and searching are based on the Presentation MathML expressions.
- Content-based search with semantic enrichment (SE): indexing and searching are based on the Content MathML expressions. The Content MathML expressions are extracted automatically using semantic enrichment module.
- Content-based search with correct Content MathML (CMathML): indexing and searching are based on the Content MathML expressions. The Content MathML expressions are those from the Wolfram Function Site.

We used the same data to train the semantic enrichment module by 10-fold cross validation method. The data is divided into 10 folds. The semantic enrichment result of each fold was done by using the other 9 folds as training data.

5.3.2 Evaluation Methodology

We used "Precision at 10" and "normalized Discounted Cumulative Gain" metrics to evaluate the results. In a large-scale search scenario, users are interested in reading the first page or the first three pages of the returned results. "Precision at 10" (P@10) has the advantage of not requiring the full set of relevant mathematical expressions, but its salient disadvantage is that it fails to incorporate consideration of the positions of the relevant expressions among the top k. In a ranked retrieval context, normalized Discounted Cumulative Gain (nDCG) as given by Equation 5.1 is a preferred metric because it incorporates the order of the retrieved expressions. In Equation 5.1, Discounted Cumulative Gain (DCG) can be calculated using the Equation 5.2, where rel_i is the graded relevance of the result at position i. Ideal DCG (IDCG) is calculable using the same equation, but IDCG uses the ideal result list which was sorted by relevance.

$$\mathrm{nDCG_p} = \frac{DCG_p}{IDCG_p} \tag{5.1}$$

$$\mathrm{DCG_p} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i)} \tag{5.2}$$

For performance analysis of the mathematical search system, we manually created 15 information needs (queries) and used them as input queries of our mathematical search system. The queries are created based on NTCIR queries with minor modification. Therefore, the search system always gets at least one exact match. Table 5.1 shows the queries we used. The top 10 results of each query were marked manually as relevant ($rel = 1$), non-relevant ($rel = 0$), or partially relevant ($rel = 0.5$). The system then calculates P@10 and an nDCG value based on the manually marked results.

Table 5.1: Queries.

No.	Query
1	$\int_0^\infty x \, \mathrm{d}x$
2	$x^2 + y^2$
3	$\int_0^\infty e^{-x^2} \, \mathrm{d}x$
4	$arcsin(x)$
5	k^2
6	$\frac{\cosh ez + \sinh ez}{e}$
7	$\mathcal{R}_z \Psi^\nu(z), \tilde{\infty}$
8	$\int \frac{a^{d+bz}}{z} \mathrm{d}z$
9	$\lim_{\nu \to \infty} \frac{L_{\alpha+\nu}}{L_\nu}$
10	$\mathcal{BP}_z \mathfrak{B}_\nu^\mu(z)$
11	$\nu \in \mathbb{N}$
12	$\Psi^\nu(z)$
13	$\log(z+1)$
14	$H_n(z)$
15	$\frac{1}{\pi} \int_0^\pi (\cos tn - z \sin t) \mathrm{d}t$

5.3.3 Experimental Results

Comparisons among the three systems were made using P@10 and nDCG scores. Table 5.2 and figure 5.5 show the P@10 and nDCG scores obtained from the search. Figure 5.4 depicts the top 10 precision of the search system. The x axis shows the k number, which ranges from 1 to 10. The y axis shows the precision score. The precision score decreased, while k increased, which indicates that the higher results are more relevant than lower results.

Table 5.2: nDCG and Precision at 10 scores of the search systems.

Method	**nDCG**	**P@10**
PMathML	0.941	0.707
CMathML	0.962	0.747
SE	0.951	0.710

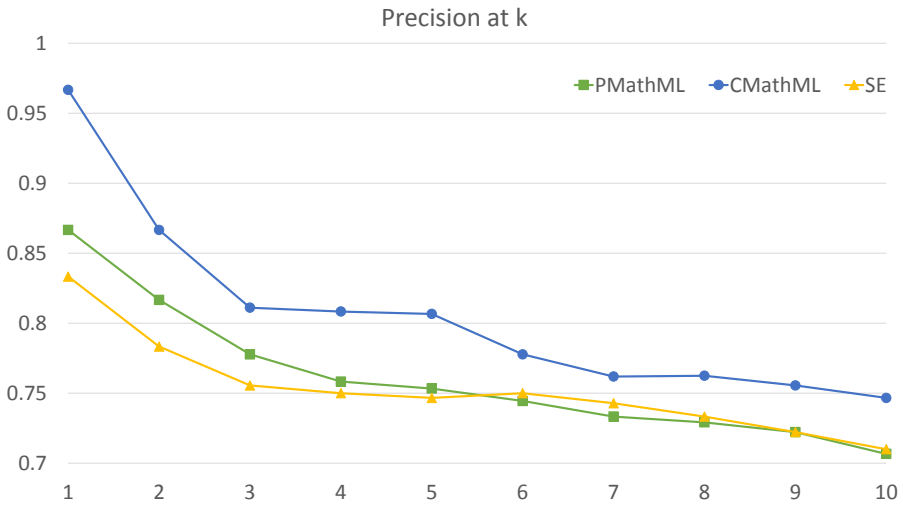

Figure 5.4: Top 10 precision of the search system.

In the experiment, a strong relation between *semantic enrichment of mathematical expressions* and *content-based mathematical search system* was found. As shown in Chapter 3, the error rate of *semantic enrichment of mathematical expressions* module is around 29 percent. With current performance, using

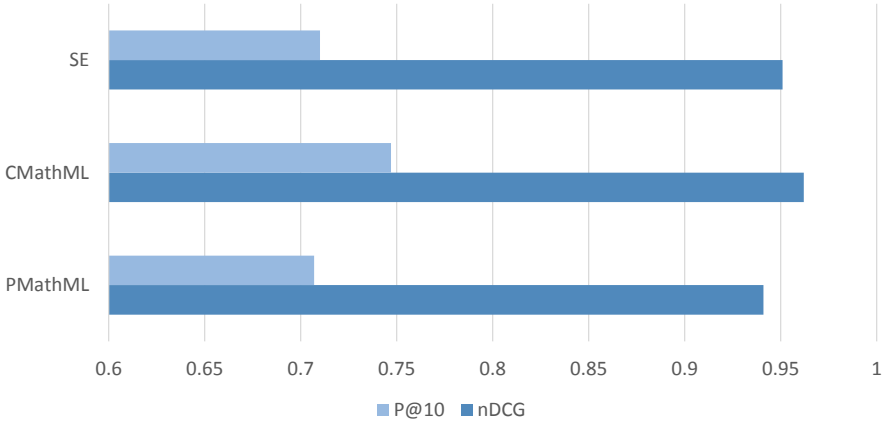

Figure 5.5: Comparison of different systems.

this module for the mathematical search system still improves the search performance. The system gained 1 percent in nDCG score and 0.3 percent in P@10 score compared to the Presentation MathML-based system. Overall, the system using perfect Content MathML yielded the highest results. In direct comparison using nDCG scores, the system using semantic enrichment is superior to the Presentation MathML-based system, although not by much. Out of 15 queries, the semantic enrichment system showed better results than Presentation MathML-based system in 7 queries, especially when the mathematical symbols contain specific meanings, e.g. Poly-Gamma function (query 10), Hermite-H function (query 14). In case the function has specific meaning but there is no ambiguity representing the function, e.g. Legendre-Q function (query 12), both systems give similar results. Presentation MathML system, however, produced better results than semantic enrichment systems in 5 queries when dealing with elementary functions (query 2, 8, 15), logarithm (query 13), and trigonometric functions (query 6) because of its simpler representation using Presentation MathML. One exception is the case of query 4, when there is more than one way to represent an expression with a specific meaning, e.g. sin^{-1} and $arcsin$, Presentation MathML

system gives unstable results.

This finding, while preliminary, suggests that we can choose either search strategy depending on the situation. We can use Presentation MathML system for elementary functions or when there is no ambiguity in the Presentation MathML expression. Otherwise, we can use a Content MathML system while dealing with functions that contain specific meanings. Another situation in which we can use a Content MathML system is when there are many ways to present an expression using Presentation MathML markup.

The average time for searching for a mathematical expression is less than one second on our Xeon 32 core 2.1 GHz 32 GB RAM server. The indexing time, however, took around one hour for 20,000 mathematical expressions. Because of the unavailability of standard corpora to evaluate content-based mathematical search systems, the evaluation at this time is quite subjective and limited. Although this study only uses 20,000 mathematical expressions for the evaluation, the preliminary experimentally obtained results indicated that the semantic enrichment approach showed promise for content-based mathematical expression search.

Chapter 6

Conclusion

This dissertation discussed the problems posed by the semantic enrichment of mathematical expressions and its application: content-based mathematical search. The semantic enrichment approach is based on statistical machine translation for translating Presentation MathML expressions into Content MathML expressions. The structural difference between Presentation and Content MathML is solved by introducing new segmentation rule. The proposed approach shows a significant improvement over a prior rule-based system. Experimental results confirm it should aid in the automatic understanding of mathematical expressions.

This dissertation also presents an approach for creating training data for the mathematical term sense disambiguation problem. Combining word-to-word alignment models and heuristic alignments, this approach shows that we can generate reasonably accurate mathematical term sense disambiguation data using available parallel corpora. The data generated can then be used to train a classifier that allows automatic sense-tagging of mathematical expressions. This study has shown that the disambiguation component using presentation features improved the system performance. The use of text features, especially the category of each expression, also played an important role in the disambiguation of

mathematical elements. The sense disambiguation module then can be incorporated with the statistical translation system to improve the overall performance of *semantic enrichment of mathematical expressions* problem. The approach, which combines statistical machine translation and disambiguation component, shows promise. Experimental results of this study showed that the proposed system achieves improvements over prior systems.

Mathematical notations are context-dependent, so to generate the correct semantic output, we must consider not just the surrounding expressions but also the document containing the notations. This dissertation considered only the first kind of context information. This being merely a first attempt at translation from Presentation to Content MathML using machine learning methods, room for improvement certainly remains. Future efforts should seek to expand the systems capacity to handle all mathematical notations. The system currently handles a limited range of mathematical notations, potential improvements for *semantic enrichment of mathematical expressions* include the following:

- Expanding training data so the system can cover more mathematical notations from different categories.
- Incorporating the information implicit in surrounding mathematical expressions; for example, definitions or other mathematical expressions.
- Improving alignment accuracy. Alignment errors can generate errors in the subsequent steps of the translation, such as rule extraction.

In contrast to natural language text, mathematical expressions require specific processing methods. More work needs to be done to establish the features best-suited to mathematical terms in a larger dataset. An extension of the model with more text and context features, in addition to the category feature, should prove interesting. Since the alignments between presentation and the content

tree affect the generated data, improving alignment accuracy may boost system performance.

This research has raised many questions in need of further investigation. One question is finding and combining new features, such as the style of the font, for the disambiguation task. Another possible improvement is making use of co-occurrence of mathematical elements in the same document. This dissertation only disambiguated lexical ambiguities of mathematical expressions. Structural ambiguities should also be considered to achieve better results. The evidence from this study suggests that in a small dataset, descriptions of mathematical expressions did not improve the system performance. Further work needs to be done to establish whether descriptions of mathematical expressions contribute to the the task in a larger dataset.

By using semantic information obtained from *semantic enrichment of mathematical expressions* module, the content-based mathematical search system has shown promising results. The experimental results confirm that this information is helpful to the mathematical search. However, this is only a first step; many important issues remain for future studies. Using an expression semantic markup is only one way of considering the semantic meaning of the formula. There are other valuable information needs to be considering as well, such as the description of the formula and its variables.

Publication list

Journal paper

- *Minh-Quoc Nghiem*, Giovanni Yoko Kristianto, and Akiko Aizawa: "Using MathML Parallel Markup Corpora for Semantic Enrichment of Mathematical Expressions", Journal of the Institute of Electronics, Information and Communication Engineers, vol.E96-D, no.8, pp. 1707-1715, August 2013.

Conference and workshop paper

- *Minh-Quoc Nghiem*, Giovanni Yoko Kristianto, Goran Topic, and Akiko Aizawa: "Sense disambiguation: from natural language words to mathematical terms", The 6th International Joint Conference on Natural Language Processing (IJCNLP 2013), Nagoya, Japan, October 2013.

- *Minh-Quoc Nghiem*, Giovanni Yoko Kristianto, Goran Topic, and Akiko Aizawa: "A hybrid approach for semantic enrichment of MathML mathematical expressions", Conferences on Intelligent Computer Mathematics (CICM 2013), Bath, United Kingdom, pp. 278-287, July 2013.

- Goran Topic, Giovanni Yoko Kristianto, *Minh-Quoc Nghiem* and Akiko Aizawa: "The MCAT Math Retrieval System for NTCIR-10 Math Track", The 10th NTCIR Conference and EVIA2013, Tokyo, Japan, pp. 680-685, June 2013.

- Giovanni Yoko Kristianto, Goran Topic, *Minh-Quoc Nghiem* and Akiko Aizawa: "Annotating Scientific Papers for Mathematical Formulae Search", Proceedings of the fifth workshop on Exploiting semantic annotations in information retrieval of The 21st ACM International Conference on Information and Knowledge Management, Hawai, USA, pp. 17-18, October 2012.

- *Minh-Quoc Nghiem*, Giovanni Yoko Kristianto, Yuichiroh Matsubayashi and Akiko Aizawa: "Automatic Approach to Understanding Mathematical Expressions Using MathML Parallel Markup Corpora", The Japanese Society for Artificial Intelligence, Yamaguchi, Japan, June 2012.

- Giovanni Yoko Kristianto, *Minh-Quoc Nghiem*, Yuichiroh Matsubayashi and Akiko Aizawa: "Extracting Definitions of Mathematical Expressions in Scientific Papers", The Japanese Society for Artificial Intelligence, Yamaguchi, Japan, June 2012.

- *Minh-Quoc Nghiem*, Giovanni Yoko Kristianto, Yuichiroh Matsubayashi and Akiko Aizawa: "Towards Mathematical Expression Understanding", Digitization and E-Inclusion in Mathematics and Science 2012, Tokyo, Japan, pp. 53-60, February 2012.

- *Minh-Quoc Nghiem*, Keisuke Yokoi, Yuichiroh Matsubayashi and Akiko Aizawa: "A Name-based Mathematical Expressions Search System", The 12th Conference of the Pacific Association for Computational Linguistics, Kuala Lumpur, Malaysia, July 2011.

- Keisuke Yokoi, *Minh-Quoc Nghiem*, Yuichiroh Matsubayashi and Akiko Aizawa: "Contextual Analysis of Mathematical Expressions for Advanced Mathematical Search", 12th International Conference on Intelligent Text

Processing and Computational Linguistics, Tokyo, Japan, pp. 81-86, February 2011.

- *Minh Nghiem Quoc*, Keisuke Yokoi and Akiko Aizawa: "Mining coreference relations between formulas and texts using Wikipedia", The Second International Workshop on NLP Challenges in the Information Explosion Era (NLPIX 2010), Beijing, China, pp. 69-74, August 2010.

- *Minh Nghiem*, Keisuke Yokoi, Akiko Aizawa: "Enhancing mathematical search with names of formulas", The Workshop on E-Inclusion in Mathematics and Science 2009, Fukuoka, Japan, pp. 22-25, December 2009.

References

Adeel, M., Cheung, H.S. & Khiyal, S.H. (2008). Math go! prototype of a content based mathematical formula search engine. *Journal of Theoretical and Applied Information Technology Vol. 4, No. 10*, 1002–1012. 1, 20, 63

Aizawa, A., Kohlhase, M. & Ounis, I. (2013). NTCIR-10 Math pilot task overview. In *National Institute of Informatics Testbeds and Community for Information access Research 10 (NTCIR-10)*, 654–661. 1, 62, 66

Altamimi, M.E. & Youssef, A.S. (2008). A math query language with an expanded set of wildcards. *Mathematics in Computer Science*, **2**, 305–331. 17, 63

Apache (2013). Apache solr. http://lucene.apache.org/solr/. 65

ASCIIMathML (2013). ASCII MathML. http://www1.chapman.edu/jipsen/mathml/asciimath.html. 10, 14

Ausbrooks, R., Buswell, S., Carlisle, D., Chavchanidze, G., Dalmas, S., Devitt, S., Diaz, A., Dooley, S., Hunter, R., Ion, P. *et al.* (2010). Mathematical markup language (MathML) version 3.0. W3C recommendation. *World Wide Web Consortium*. 10, 23, 63

Batchelor, C.R. & Corbett, P.T. (2007). Semantic enrichment of journal articles using chemical named entity recognition. In *Proceedings of the 45th*

Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, ACL '07, 45–48, Association for Computational Linguistics, Stroudsburg, PA, USA. 4

Berners-Lee, T. (2000). *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web*. HarperBusiness. 4

Birch, A. & Osborne, M. (2011). Reordering metrics for mt. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 1027–1035, Association for Computational Linguistics, Portland, Oregon, USA. 24

Bird, S., Dale, R., Dorr, B., Gibson, B., Joseph, M., Kan, M.Y., Lee, D., Powley, B., Radev, D. & Tan, Y.F. (2008). The acl anthology reference corpus: A reference dataset for bibliographic research in computational linguistics. In *Language Resources and Evaluation Conference (LREC 08)*. 3, 24, 35

Brown, P.F., Cocke, J., Pietra, S.A.D., Pietra, V.J.D., Jelinek, F., Lafferty, J.D., Mercer, R.L. & Roossin, P.S. (1990). A statistical approach to machine translation. *Computational Linguistics Volume 16 Issue 2*, 79–85. 15, 23

Brown, P.F., Pietra, V.J.D., Pietra, S.A.D. & Mercer, R.L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics V. 19 Issue 2*, 263–312. 15

Buswell, S., Caprotti, O., Carlisle, D.P., Dewar, M.C., Gaetano, M. & Kohlhase, M. (2004). The openmath standard. Tech. rep., version 2.0. The Open Math Society, 2004. 9, 63

Carpuat, M. & Wu, D. (2007). Improving statistical machine translation using word sense disambiguation. In *In The 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007*, 61–72. 16

Cavnar, W.B. & Trenkle, J.M. (1994). N-gram-based text categorization. In *In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, 161–175. 53

Chan, Y.S. & Ng, H.T. (2005). Scaling up word sense disambiguation via parallel texts. In *Proceedings of the 20th national conference on Artificial intelligence - Volume 3*, 1037–1042. 16

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 263–270. 15

Cornell University Library (2013). arxiv. http://arxiv.org/. 66

Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, **20**, 273–297. 48

Diab, M. & Resnik, P. (2002). An unsupervised method for word sense tagging using parallel corpora. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 255–262. 16

Gao, Q. & Vogel, S. (2008). Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, SETQA-NLP '08, 49–57, Association for Computational Linguistics, Stroudsburg, PA, USA. 15

Ginev, D., Jucovschi, C., Anca, S., Grigore, M., David, C. & Kohlhase, M. (2009). An architecture for linguistic and semantic analysis on the arXMLiv corpus. In *Applications of Semantic Technologies*. 14

Grigore, M., Wolska, M. & Kohlhase, M. (2009). Towards context-based disambiguation of mathematical expressions. In *The Joint Conference of ASCM 2009 and MACIS 2009: Asian Symposium on Computer Mathematics and Mathematical Aspects of Computer and Information Sciences*, 262–271. 13

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. & Witten, I.H. (2009). The weka data mining software: an update. *SIGKDD Explor. Newsl.*, **11**, 10–18. 56

Ide, N., Erjavec, T. & Tufis, D. (2001). Automatic sense tagging using parallel corpora. In *In Proceedings of the 6 th Natural Language Processing Pacific Rim Symposium*, 212–219. 17

Ide, N., Erjavec, T. & Tufis, D. (2002). Sense discrimination with parallel corpora. In *Proceedings of the ACL-02 workshop on Word sense disambiguation: recent successes and future directions-Volume 8*, 61–66. 17, 43

Kamali, S. & Tompa, F.W. (2009). Improving mathematics retrieval. In *2nd Workshop Towards a Digital Mathematics Library*, 37–48. 64

Kamali, S. & Tompa, F.W. (2013). Structural similarity search for mathematics retrieval. In *MKM/Calculemus/DML*, 246–262. 64

Kan, M.Y. (2013). Association for computational linguistics anthology reference corpus. http://acl-arc.comp.nus.edu.sg/. 3, 24, 35, 57

Kernighan, B.W. & Cherry, L.L. (1975). A system for typesetting mathematics. *Communications of the ACM, Volume 18 Issue 3*, 151–157. 12

KNUTH, D.E. (1984). *The TEXbook*. Addison-Wesley. 8

KOEHN, P., OCH, F.J. & MARCU, D. (2003). Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, 48–54, Association for Computational Linguistics, Stroudsburg, PA, USA. 39

KOHLHASE, M. (2006). Omdoc an open markup format for mathematical documents version 1.2. In *Number 4180 in Lecture Notes in Artificial Intelligence*, Springer Verlag. 9

KOHLHASE, M. & PRODESCU, C.C. (2013). Mathwebsearch at NTCIR-10. In *National Institute of Informatics Testbeds and Community for Information access Research 10 (NTCIR-10)*, 675–679. 1, 20, 63

KOHLHASE, M. & SUCAN, I.A. (2006). A search engine for mathematical formulae. *Artificial Intelligence and Symbolic Computation Lecture Notes in Computer Science Vol. 4120*, 241–253. 1, 20, 63

KRISTIANTO, G.Y., TOPIC, G., NGHIEM, M.Q. & AIZAWA, A. (2012). Annotating scientific papers for mathematical formula search. In *Proceedings of the Fifth workshop on Exploiting Semantic Annotations in Information Retrieval*, 17–18. 58

LAMPORT, L. (1986). *LATEX: A Document Preparation System*. Addison-Wesley. 8

LEFEVER, E. & HOSTE, V. (2010). Semeval-2010 task 3: Cross-lingual word sense disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, 15–20. 17

Lefever, E., Hoste, V. & De Cock, M. (2011). Parasense or how to use parallel corpora for word sense disambiguation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 317–322. 17

Liang, P., Taskar, B. & Klein, D. (2006). Alignment by agreement. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, 104–111, Association for Computational Linguistics, Stroudsburg, PA, USA. 15

Liska, M., Sojka, P. & Ruzicka, M. (2013). Similarity search for mathematics: Masaryk University Team at the NTCIR-10 math task. In *National Institute of Informatics Testbeds and Community for Information access Research 10 (NTCIR-10)*, 686–691. 1

MathOverflow (2013). Math overflow. http://mathoverflow.net/. 21

MathsAssess (2013). Maths assess project. http://mathassess.ecs.soton.ac.uk/. 23

McKain, D. (2013). SnuggleTeX version 1.2.2. http://www2.ph.ed.ac.uk/snuggletex/. 1, 2, 4, 13, 23, 25, 47, 63

Miller, B.R. (2013). LaTeXML a LaTeX to XML converter. http://dlmf.nist.gov/LaTeXML/. 1, 2, 14, 23, 47, 63

Miller, B.R. & Youssef, A.S. (2003). Technical aspects of the digital library of mathematical functions. *Annals of Mathematics and Artificial Intelligence*, **38**, 121–136. 18

Miner, R. & Munavalli, R. (2007). An approach to mathematical search through query formulation and data normalization. In *Towards Mechanized Mathematical Assistants*, vol. 4573 of *Lecture Notes in Computer Science*, 342–355, Springer Berlin Heidelberg. 17, 63

Miutka, J. & Galambo, L. (2011). System description: Egomath2 as a tool for mathematical searching on wikipedia.org. In J. Davenport, W. Farmer, J. Urban & F. Rabe, eds., *Intelligent Computer Mathematics*, vol. 6824 of *Lecture Notes in Computer Science*, 307–309, Springer Berlin Heidelberg. 19

Munavalli, R. & Miner, R. (2006). Mathfind: a math-aware search engine. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 735–735, ACM. 18

Nagata, M., Saito, K., Yamamoto, K. & Ohashi, K. (2006). A clustered global phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, 713–720, Association for Computational Linguistics, Sydney, Australia. 24

National Institute of Standards and Technology (2013). Digital library of mathematical functions. http://dlmf.nist.gov. 17, 18, 63

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Computing Surveys*, **41**, 1–69. 43

Nghiem, M.Q., Kristianto, G.Y. & Aizawa, A. (2013a). Using mathml parallel markup corpora for semantic enrichment of mathematical expressions. *Journal of the Institute of Electronics, Information and Communication Engineers, vol.E96-D, no.8*, **69**, 1707–1715. 5, 22, 47

Nghiem, M.Q., Kristianto, G.Y., Topic, G. & Aizawa, A. (2013b). A hybrid approach for semantic enrichment of mathml mathematical expressions. In *In Proceedings of the Conferences on Intelligent Computer Mathematics (CICM 2013)*, 278–287. 5, 42

Nghiem, M.Q., Kristianto, G.Y., Topic, G. & Aizawa, A. (2013c). Sense disambiguation: from natural language words to mathematical terms. In *In Proceedings of the 6th International Joint Conference on Natural Language Processing (IJCNLP 2013)*. 5, 42

Nguyen, T.T., Chang, K. & Hui, S.C. (2012). A math-aware search engine for math question answering system. In *Proceedings of the 21st ACM international conference on Information and knowledge management (CIKM 2012)*, 724–733. 1, 21, 63

Och, F.J. & Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics Volume 29 Issue 1*, 19–51. 15, 28, 44, 50

Oracle (2013). Apache OpenOffice Math. http://www.openoffice.org/product/math.html. 12

Padó, S. & Lapata, M. (2009). Cross-lingual annotation projection for semantic roles. *Journal of Artificial Intelligence Research*, **36**, 307–340. 17

Peñas, A. & Hovy, E. (2010). Semantic enrichment of text with background knowledge. In *Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading*, FAM-LbR '10, 15–23, Association for Computational Linguistics, Stroudsburg, PA, USA. 4

Schafer, U., Read, J. & Oepen, S. (2012). Towards an ACL anthology corpus with logical document structure. An overview of the ACL 2012 contributed

task. In *Proceedings of the ACL-2012 Special Workshop on Rediscovering 50 Years of Discoveries*, 88–97. 35

Siekmann, J. ((visited on 01 March. 2014)). Activemath. http://www.activemath.org/eu/. 19

Snover, M., Dorr, B., Schwartz, R., Micciulla, L. & Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, 223–231. 36, 58

Sojka, P. & Líška, M. (2011). The Art of Mathematics Retrieval. In *Proceedings of the ACM Conference on Document Engineering, DocEng 2011*, 57–60, Association of Computing Machinery, Mountain View, CA. 19

Springer (2013). Springer LaTeX Search. http://www.latexsearch.com/. 17, 63

Stamerjohanns, H., Ginev, D., David, C., Misev, D., Zamdzhiev, V. & Kohlhase, M. (2009). Mathml-aware article conversion from latex. In *DML 2009: Proceedings of the 2nd workshop*, 109–120. 12

Sun, J., Zhang, M. & Tan, C.L. (2010). Exploring syntactic structural features for sub-tree alignment using bilingual tree kernels. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 306–315. 44

Suzuki, M., Kanahori, T., Ohtake, N. & Yamaguchi, K. (2004). An integrated OCR software for mathematical documents and its output with accessibility. *Computers Helping People with Special Needs, Lecture Notes in Computer Science Volume 3118*, 648–655. 7

Tinsley, J., Zhechev, V., Hearne, M. & Way, A. (2007). Robust language pair-independent sub-tree alignment. In *In Proceedings of MT Summit XI -07*. 46

Topic, G., Kristianto, G.Y., Nghiem, M.Q. & Aizawa, A. (2013). The MCAT math retrieval system for NTCIR-10 Math track. In *National Institute of Informatics Testbeds and Community for Information access Research 10 (NTCIR-10)*, 680–685. 65, 67

Tufiş, D., Ion, R. & Ide, N. (2004). Fine-grained word sense disambiguation based on parallel corpora, word alignment, word clustering and aligned wordnets. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04. 17

Uniquation (2013). Uniquation search engine. http://uniquation.com/. 17

Wolfram (2013). The Wolfram Functions Site. http://functions.wolfram.com/. 1, 3, 19, 24, 34, 57, 63, 66

Wolska, M. & Grigore, M. (2010). Symbol declarations in mathematical writing. In *Towards a Digital Mathematics Library. Paris, France, July 7-8th, 2010*, 119–127. 13, 14

Wolska, M., Grigore, M. & Kohlhase, M. (2011). Using discourse context to interpret object-denoting mathematical expressions. In *Towards a Digital Mathematics Library. Bertinoro, Italy, July 20-21st, 2011*, 85–101. 13, 14, 43

Wong, Y.W. & Mooney, R. (2006). Learning for semantic parsing with statistical machine translation. In *Proceedings of the 2006 Human Language Technology Conference - North American Chapter of the Association for Computational*, 439 – 446. 16

Yamada, K. & Knight, K. (2001). A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association for Computational*, 523–530. 15

Yang, N., Li, M., Zhang, D. & Yu, N. (2012). A ranking-based approach to word reordering for statistical machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 912–920, Association for Computational Linguistics, Jeju Island, Korea. 24

Yokoi, K. & Aizawa, A. (2009). An approach to similarity search for mathematical expressions using mathml. In *2nd Workshop Towards a Digital Mathematics Library*, 27–35, DML 2009. 17, 20, 63

Youssef, A.S. (2005). Information search and retrieval of mathematical contents: Issues and methods. In *The ISCA 14th International Conference on Intelligent and Adaptive Systems and Software Engineering*, 100–105. 17, 18, 63

Youssef, A.S. (2007). Methods of relevance ranking and hit-content generation in math search. In *Towards Mechanized Mathematical Assistants*, 393–406. 19

Youssef, A.S. & Altamimi, M.E. (2007). An extensive math query language. In *SEDE*, 57–63. 17, 63

Zhang, K. & Shasha, D. (1989). Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing, Volume 18 Issue 6*, 1245–1262. 36