Identifying suitable mathematical translation candidates from the logs of Dr. Math

Bertram Haskins
School of Information and Communication Technology
Nelson Mandela Metropolitan University
Port Elizabeth, South Africa, 6001
Email: Bertram.Haskins@nmmu.ac.za

Reinhardt A Botha
School of Information and Communication Technology
Nelson Mandela Metropolitan University
Port Elizabeth, South Africa, 6001
Email: ReinhardtA.Botha@nmmu.ac.za

***Abstract*—Dr. Math is a service, which connects high school students with math problems to volunteer human tutors. Some of the tutors on the Dr. Math service have difficulty in servicing queries received in Mxit lingo. Identifying which of these queries contain valid mathematical questions steals time which could be better spent on the actual tutoring process. This paper develops and tests filtering algorithms based on numbers, symbols and tag words, in order to identify queries containing suitable mathematical translation candidates. A combination of numeric and symbolic filtering yields the most accurate results, whereas filtering using numbers, symbols and tag words returns the highest number of results. On average, the algorithms return their filtered results in under a millisecond.**

***Index Terms*—Text analysis**

I. Introduction

i wnt u 2 hlp me wth maths abt mxid frections

Does the statement above have you scratching your head? It's an example of how student queries generally start in Dr. Math. If the initial question is so ill-phrased, imagine having to translate statements written in this shorthand to usable mathematical equations.

simp : sq rt 27 . sq rt 18 . sq rt 32 ova sq rt 12 . sq rt 8

With enough time and experience, a tutor should be able to decipher statements such as the one above. But what if they didn't have to?

A. Mxit and Dr. Math

Mxit is an on-line chat service, mainly used on mobile telephones. It provides similar functionality to other services such as Google Talk, but with added features, such as chat rooms, games and apps. Mxit has been broadly adopted by South African school learners, because of it's relatively low usage cost and it's availability on handsets from most mobile telephone manufacturers.

Mxit lingo is a general term for the non-standardized, shorthand language used by teenagers when communicating on the social platform Mxit. One of the services available on Mxit is a a math tutoring service, called Dr. Math, which has been created to take advantage of the large user base that Mxit has under South African school learners. Dr. Math currently allows over 30 000 learners to query volunteer human tutors with mathematical queries [1].

B. Problem statement and paper objective

Some tutors on the Dr. Math service have difficulty in reading the Mxit lingo statements received during tutoring sessions. Attempting to decipher the messages wastes time which could be more productively spent in tutoring other school learners. Implementing a system to automatically render qualifying queries as well-formatted mathematical equations may support the tutors, by allowing them to focus on relevant queries without first attempting to perform a translation.

This paper addresses the problem of identifying queries which are candidates for translation to mathematical equations. Thus, the objective of this paper is to devise a method with which to sift through Mxit lingo queries to determine which statements may be valid candidates for translation to a mathematical equation.

II. Methodology

To meet the paper's objective, the design science research methodology [2] has been followed. The activities of the methodology are shown in Figure 1.

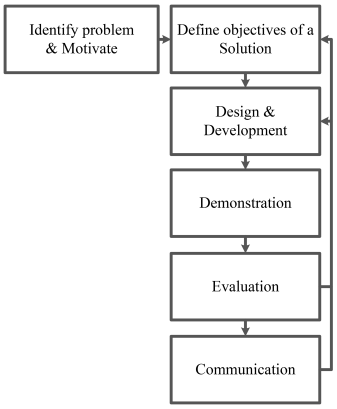

Figure 1. The design science research methodology activities [2].

In design science an artifact may be a construct, model, method or instantiation that clarifies or solves problems in implementing successful information systems [3]. This study will deliver algorithms to filter out suitable mathematical translation candidates, from the logs of Dr. Math, as its artifact.

The first two activities *identify problem* and *motivate and define objectives of a solution*, have been completed by defining the initial problem statement and research objective.

The *design and development* activity will involve the development of the algorithms (the artifact) necessary for providing the final solution to the problem. The fourth activity, *demonstration*, requires that the artifact be used to solve an instance of the problem. This will be done by means of using the artifact on a test data set, to gather experimental results. The artifact will be *evaluated* by comparing the results of testing the artifact with the initial objective of the solution

This paper forms part of the *Communication* activity, which states that the problem, its importance and its artifact must be shared with with a relevant and applicable audience. The design science research methodology requires iteration through previous activities to ensure that the artifact sufficiently solves the research problem. Previous activities will be revisited after performing initial tests using training data. The results from these tests will be used to modify the artifact to more accurately function within the problem domain, before retesting the artifact on the training set. The artifact will then be further tested on a test data set.

The Council for Scientific and Industrial Research (CSIR) Meraka Institute are responsible for the development of Dr. Math. All the processing and testing of the developed artifact will be done on the historic system logs of the Dr. Math service. The Dr. Math logs were obtained, with permission, from the CSIR.

The following sections will detail the steps followed in the development and testing of the artifact, by discussing the various filtering algorithms in turn. An overview of the steps involved in this process is shown in Figure 2.

III. Log filtering

In order for a log filtering process to be feasible, the process needs to be completely automated. Tutors deal with multiple students concurrently during tutoring sessions and may receive multiple queries from each of these learners.

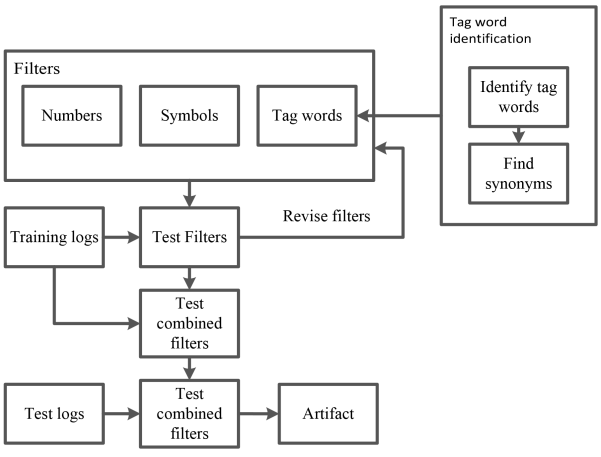

Figure 2. Steps followed in the development and testing of the artifact

Table I
Tag descriptors

Tag	Description
N	The entry's topic is not mathematical in nature.
Y	The entry's topic is mathematical in nature, but not a suitable translation candidate.
X	The entry's topic is mathematical in nature and is a suitable translation candidate.

One of the problems with the tutoring process is finding willing, suitable tutors. Therefore, appointing extra personnel to perform filtering on logs, before passing them along to the tutors, is not feasible.

Since the automated process has to emulate the judgement of a human tutor in order select suitable translation candidates, it was necessary to create tagged versions of the logs. The Dr. Math logs represent the anonymous conversations between tutors and learners. For the purposes of this paper, the flow of a conversation is not tracked, instead the contents of individual messages, forming part of a conversation, are analyzed. To facilitate the tagging process, the logs were split into groupings of 500 entries (messages) each. Five of these groupings were randomly selected to act as a training data set and a further five were selected to act as the final testing set.

The tagging process was simplified by developing an application, which displays all 500 entries for a given group, in sequence. Each entry was then individually scrutinized (without regard for preceding or following entries) and tagged as shown in Table I.

Scrutiny of the Dr. Math logs reveal that learners phrase their mathematical queries using either numbers, symbols (non-alphanumeric characters), words or a combination of all three concepts. Therefore, the decision was made to attempt to create filters to detect suitable translation candidates, based on these three concepts. The following sections describe the processes and motivations involved in performing log filtering by numbers, symbols or tag words. The rules created and discussed in the following sections were informed by studying a sampling of South African high school mathematics text books [4], [5], [6], [7], [8].

IV. Numeric filtering

In order to filter on numbers, every log entry was checked to see whether it contained the numeric characters *0* to *9*. This was thought to be an easy indicator as to whether or not a log entry pertained to a mathematical topic and whether it would be possible to translate the query to a mathematical equation. However, upon scrutiny, the contents of the log entries prohibited this from being a viable solution for three reasons:

1) A log entry may contain the on-line pseudonym of a learner. These names sometimes contain a number, which may or may not be separated form the alphabetic portion of the name by means of a space or non-alphanumeric character.

2) Some learners discuss events in their personal lives with the tutors. These topic include their placement in an athletic event or competitions. This results in entries such as *1st* or *4th*, which could be identified as being an indicator of a mathematical topic when filtering on numbers.
3) Learners use numbers to provide typing shortcuts for certain words, e.g. *2* instead of *to* and *l8r* instead of *later*.

To decrease the number of false positives, the following set of rules was identified for the selection of translatable content by means of number filtering:

1) An entry must contain at least two instances of numeric character strings. This decision was made to ensure that there are at least 2 separate number sequences being applied to one another by means of an operator. This, however, has the detrimental effect of eliminating entries such as *5z* or *50 + a* from being viable translation candidates.
2) A numeric string may not be preceded or followed by more than 3 non-alphanumeric characters. Our study of equations in South African high school mathematics textbooks show very few entries for equations with more than 3 concurrent non-alphanumeric symbols.
3) A numeric string may not be directly preceded by an alphabetic character. Most equations in South African high school textbooks are written in the format of a constant followed by a variable, i.e. *3a* and rarely *a3*.
4) A numeric string may not be directly followed by more than 3 alphabetic characters. Our study of equations in South African high school mathematics textbooks indicate that formulas rarely contain more than three variables per grouping, i.e. *3abc* and very rarely *3abcd*.

Table II displays the results of applying the rules to the training data set. The addition of preprocessing and the introduction of other rule sets yielded no improvement on results. As such, the initial rule set was kept unchanged.

The *Manual* column indicates how many entries were identified as being suitable for translation under human scrutiny. The *Filtered* column indicates how many translation candidates the filtering process discovered, without regard for whether they are actual candidates or not. The *% Found* indicates how large a percentage of the manually identified candidates were identified by the filtering process. The last column *Accuracy %* specifies the percentage of how many of the candidates identified by the filter are valid candidates.

Table II
TRAINING RESULTS FOR NUMERIC FILTERING

Set	Manual	Filtered	% Found	Accuracy %
Training set 1	81	3	2.47	66.67
Training set 2	53	3	5.66	100.00
Training set 3	33	0	0.00	100.00
Training set 4	45	3	2.22	33.34
Training set 5	21	2	0.00	0.00
Averages:	47	2	2.07	60.00

Table III
TRAINING RESULTS FOR SYMBOL-BASED FILTERING

Set	Manual	Filtered	% Found	Accuracy %
Training set 1	81	57	61.73	87.72
Training set 2	53	26	35.85	73.08
Training set 3	33	14	21.21	50.00
Training set 4	45	42	57.78	61.90
Training set 5	21	53	57.14	22.64
Averages:	47	38	46.74	59.07

V. FILTERING ON SYMBOLS

For the purpose of filtering on symbols, a set of non-alphanumeric characters were identified. In order to qualify for selection, these characters had to fulfill certain prerequisites:

1) They had to represent a specific mathematical operator or construct as found in South African high school mathematics textbooks.
2) A school learner must be able to type the character using a mobile phone keypad.

A set of rules was then created in order to identify entries which may be candidates for translation, based on the symbols they contain. The rules are as follows:

1) An entry has to contain at least one instance of one of the identified characters.
2) An entry is disqualified if it contains more than 3 consecutive instances of these identified characters. Some of these symbolic characters may be used by students in their on-line pseudonyms. Mathematical equations in South African high school textbooks rarely contain more than three consecutive symbols.
3) An entry is disqualified if it contains more than 3 consecutive alphabetic preceding or following characters. A study of equations in a sampling of South African high school mathematics textbooks indicate that formulas rarely contain more than three variables per grouping, i.e. *+abc* or *abc+* might occur, but not *+abcd* or *abcd+*.

After scrutinizing the results of performing symbol-based filtering on the training set (Table III), it became clear that the following factors were influencing the accuracy of the results:

1) There are multiple instances in the logs where learners attempt to gather automatic responses. Auto-responses provide functionality such as an encyclopedia look-up. The auto-response commands are always in the form of a period followed by an alphabetic character or two. In some cases a period may be used to indicate multiplication and the alphabetic characters following may be interpreted as variables. The decision was made to remove all these automatic response commands, by means of preprocessing.
2) Emoticons are collections of symbols, such as *:)*, used to convey emotions. These emoticons may consist of valid symbolic characters such as ")" or "(". Even though we were aware of their existence beforehand, we did

Table IV
REPROCESSED TRAINING RESULTS FOR SYMBOL-BASED FILTERING

Set	Manual	Filtered	% Found	Accuracy %
Test set 1	81	75	81.48	88.0
Test set 2	53	46	66.04	76.09
Test set 3	33	30	66.67	73.33
Test set 4	45	47	82.22	78.72
Test set 5	21	64	80.95	26.56
Averages:	47	38	75.47	68.54

not simply want to filter all of them out, because they may not have an effect on the identification process. The decision was made to remove any emoticons, containing symbols used in mathematical equations and identified in the training set, by means of preprocessing.

3) Most of the equations identified in the training set contained the operators plus, minus, divide, multiply and the equal sign. A rule was added to give extra weighting to any entry which contained these specific symbols.

Table IV demonstrates that the *% Found* shows an increase of 28.73% and the *Accuracy %* an increase of 9.47% after the introduction of preprocessing and the additional rule.

VI. SELECTING TAG WORDS

Filtering the queries for tag words depend on two basic principles:

1) Some learners type out equations using words, instead of symbols. This may be because of preference or lack of knowledge as to which symbol, on a mobile telephone keypad, to use for concepts such as exponents or fractions. In cases such as these there may be certain words, such as plus or minus, which could be directly construed as indicators for the presence of mathematical equations.
2) In other cases however, there may be words which are not used by learners to type a mathematical equation, but they may form part of a question or word sum which may still be translated to a mathematical equation.

In order to satisfy the needs of both these principles, a study was done to identify which English words are prevalent in the South African high school mathematics curriculum.

The first phase of the study involved manually scrutinizing South African high school mathematics textbooks [4], [5], [6], [7], [8] and typing any statements which form part of either a question or an explanation of an answer into a custom application. This application enabled the identification of distinct words from these textbooks as well as a counter for how often they occur.

The second phase of the study was performed by the creation of software to identify and count instances of individual words from an input file. The South African curriculum statements for mathematics [9] and mathematics literacy [10] were used as input to this software. The curriculum statements do not contain only text, but examples of equations as well. The software captured these equations as individual words, which skewed the results.

The results from the first and second phase were combined to form a single list of words and their associated rates of occurrence.

Because this list still contained some equations, identified in phase two, a free on-line English dictionary [11] was sourced and converted to a compatible format. The words in the combined list were compared to the entries in the dictionary and any illegal entries, such as equations and incorrectly spelled words were filtered out. After the filtering process, 244 words remained of which some were plural forms. After removing these plural forms a final tally of 233 individual words remained.

If this paper focused on the selection of translation candidates from well-formed English statements to mathematical equations, these identified words may have been sufficient, but because Mxit lingo contains a non-standardized form of English, a last phase was necessary in order to identify possible synonyms for these words from the historic logs of Dr. Math. Performing this process manually would be prohibitively time-consuming, so a decision was made to automate the initial selection of synonyms and then perform a final manual selection from the results of the automated selection.

For the purpose of automatically selecting synonyms for the 233 words, custom software was written to analyze an input set of Dr. Math logs. Several techniques were then used to process these logs to attempt to identify synonyms. The following sections discuss the various techniques used.

A. Containment

The simplest method of detecting word similarity is to test wether one word contains the other. If the one word is already in its root form, then the comparison yields a low-cost and efficient means of detecting word similarity.

B. Weighting

By supplementing the containment process discussed in the previous section, with weighting, the process becomes a bit more accurate. The weighting process works on the principle that whenever one word contains another, there might be some letters left over. The less letters left, the greater the chance that the words are related. If there are no letters left over, the match would be 100%. If one word is contained by another, the initial weighting is 40%. The value of 40 % was chosen by means of trial and error. Table V shows which percentage is added to the match for certain length differences between words.

C. Stemming

There are a variety of stemming algorithms (stemmers), which are used to group words based on semantic similarity [12]. Stemmers change words by either removing pre- and suffixes or by substituting them, e.g. *engineering* is changed to *engineer*. A stemming algorithm is a computational procedure which reduces all words with the same root (or, if prefixes are

Table V
WEIGHTING AS APPLIED TO WORD LENGTH DIFFERENCE

Difference (in letters)	Added %	Total Match %
1	50	90
2	30	70
3	20	60
4	10	50
5	5	45
6	1	41

left untouched, the same stem) to a common form, usually by stripping each word of its derivational and inflectional suffixes [13].

The premise behind stemming is to get a word as close to its root English form as possible. This lessens the amount of words necessary in the normalized text base, which in turn lessens the amount of comparisons necessary to facilitate an accurate translation.

Stemming algorithms generally try to match the longest possible affix to one stored in a list. Once this is complete the algorithm will try to handle any spelling differences between root forms. One of the earliest stemming algorithms was proposed by Lovins [13]. This stemmer contains 294 endings, 29 conditions and 35 transformation rules. Various implementations of the stemmer modify these settings or add their own depending on the target language.

The stemmer chosen for this study was developed by Porter [14]. The original Porter stemmer consisted of 5 steps, each consisting of various rules, which are tested in turn. These rules may be adjusted to fit certain applications or languages, but their intention remains to determine whether two words,*W1* and *W2*, may be reduced to a common stem *S*, while retaining the meaning of their parent sentences. Many applications have taken to using the default set of rules provided by the stemming algorithm, without optimization. To this end, Porter has provided a free software implementation of his algorithm. This study makes use of a Visual C# variant of the algorithm.

The approach followed by Porter does however share some common ground with Lovins, in that they both describe a general algorithm for stemming and that they provide a specific collection of rules under which the algorithm may be applied [15].

D. Partial stemming

Partial stemming is the process of removing known prefixes or suffixes, but not in conjunction. In order for this process to work, a list of predefined prefixes and suffixes was created. A list of 38 prefixes and 37 widely used English suffixes, ranging in length from 1 to 5 letters, were compiled. Word comparisons were then done by attempting to remove these affixes from a target word, in turn, and then attempting to match the modified word to the input word.

E. Adjacency

Using adjacency to determine word similarity works on the premise that words serving a similar purpose should routinely be surrounded by the same words. An adjacency lists is generated for a word by determining how many times a word appears in conjunction with a given word in relation to all of the other words in the corpus of a given text-base or language. If an adjacency list is generated for the word *plus*, it may reflect adjacencies to words such as *five*, *ten* and *twenty*. If an adjacency list is generated for the word *add*, it is to be expected that it may occur adjacent to these same words at some stage. By comparing how many words out of the total corpus of words in the language both *plus* and *add* are adjacent to, we arrive at a matchable percentage. The higher this percentage, the greater the chance that a word may be considered a synonym for another.

F. Word dilution

Word dilution refers to the process of replacing all double occurrences of a letter in a word with a single occurrence of the letter, i.e. *address* gets converted to *adres*. This is a lightweight means of identifying words which might have been misspelled in a given sentence. In this study, when a word has been converted in this way, it is said to be in its base form. This should not be confused with the root form of a word, which is the result of applying a stemming algorithm to a word.

The author has not been able to find any precedence for this procedure, by means of literature study, but has been using these rules for synonym / substitute matching for a few years. This process is incorporated into a proprietary piece of software, called EasyMark [16], which is used for the automatic marking of student scripts.

Depending on how misspelled the word is, the dilution may need to be taken a step further, by removing the vowels from the interior of the word, as well. In such an example *address* would get converted to *adrs*. Vowels at the start or end of the word are usually not removed, as they tend to be placed correctly. This results from people typing words as they say and hear them. The beginnings and endings of words generally tend to be pronounced very distinctively, so most spelling errors are usually made in the middle of words. Medial letters of a word have more neighbours than letters at the periphery of a word, so they are more prone to being misspelled [17].

G. N-grams

Any word can be divided into smaller chunks. The smallest possible chunks being single letters. Inherently humans divide words into syllables, but a computing algorithm would not need to divide a word using these same principles. N-grams are a means of dividing a word into smaller overlapping chunks. These individual chunks could then be compared to the chunks of another word to determine similarity.

The letter *n* in the word n-gram refers to the variability in the length of the individual word chunks. Different applications may use different lengths of n-grams to different effects. Some approaches even combine several different lengths simultaneously or append blanks to the beginning and ending of a word. This helps with matching beginning-of-word and ending-of-word situations [18].

Table VI
TRAINING RESULTS FOR TAG WORD-BASED FILTERING

Set	Manual	Filtered	% Found	Accuracy %
Training set 1	81	9	7.41	66.67
Training set 2	53	4	5.66	75.00
Training set 3	33	19	27.27	47.37
Training set 4	45	18	11.11	27.78
Training set 5	21	6	14.29	50.00
Averages:	47	2	13.15	53.36

According to [19], character n-gram tokenization is an attractive alternative to stemming. Some of the n-grams derived from a word will span only portions of the word which do not show any differentiation from the word's root form. This means that many of the benefits of stemming can be achieved without any knowledge of the target language. This study employs uni-, bi- and trigrams.

VII. FILTERING ON TAG WORDS

The techniques discussed in the previous section were all employed on every one of the 233 identified tag words. An average percentage match was generated for each word with regards to each of the words contained in the input logs. The top 50 automated synonym matches for each word was then stored in a separate file. These 233 files (each containing 50 entries) were then manually scrutinized to determine which of the identified words may be seen as synonyms if encountered by a human.

These synonyms included plurals, words sharing a common root, misspelled words and completely different words. Finally, these words were compiled into a single list and filtered for duplicates, resulting in a final total of 1775 tag words.

The following set of rules were compiled in order to facilitate the process of selecting translation candidates by filtering on tag words.

1) A query must contain at least two of the tag words in order to be considered a candidate for translation. Some of the tag words may appear in normal conversation, but in a different context. Thus the decision was made to have at least two of these words in a query, to attempt to rule out general conversational usage.
2) Extra weight is added to a query if it contains predefined word pairs, which may be indicators of alternate means of specifying mathematical operations. A list of these words were sourced from [20].

Table VI show the results of performing filtering on the training set using tag word filtering. The addition of preprocessing and testing various other rule sets did not yield any improvements to the technique. As such, the initial rule set was kept unchanged.

VIII. COMPARING AND COMBINING

Having selected and implemented the three different filtering techniques, we decided to compare which of the techniques

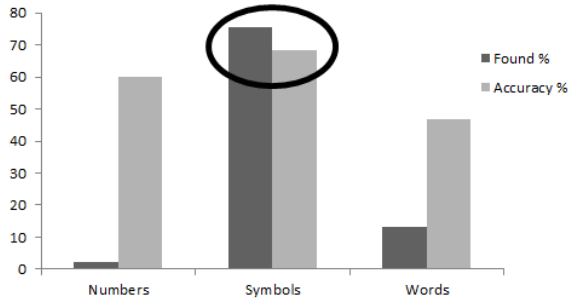

Figure 3. Comparison of the three filtering techniques

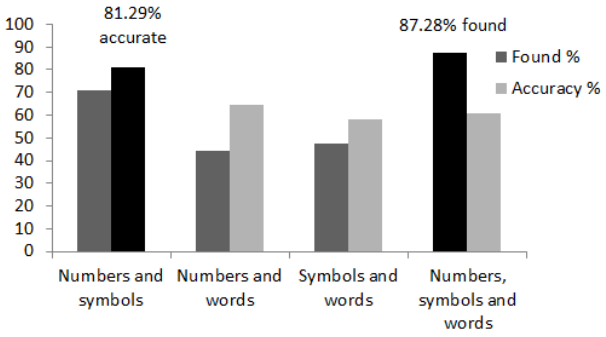

Figure 4. Results obtained from combining the filtering techniques

yielded the best results. This comparison is shown in Figure 3.

The results show a clear difference in both the number of results found and accuracy of the three filtering techniques. Filtering on symbols yield the most results, as well as the most accurate results. A problem with comparing the three techniques directly are that they should find completely different candidates, except if by chance a query contains elements to trigger for more than one of the techniques.

Most queries from learners contain some combination of numbers, symbols and words. In order to test whether combining the filters yielded different results, the training set was reprocessed using various combinations of the three filters. The combination tests also measured the average length of time processing takes for each query, to ensure that the processes are feasible in real-time. Even combining all three filters only yielded an average processing time of 0.84 milliseconds. This processing time will vary, depending on which hardware platform the processing is performed on, but serves to indicate that the filtering algorithms provide results in real-time.

As Figure 4 illustrates, a combination of filtering on numbers, symbols and tag words, returns the highest number of translation candidates, but does not return the least false positives. The most accurate of the processes are a combination of filtering on numbers and symbols.

Figure 5 demonstrates that the filters are not only applicable to the training set, by applying the three combined filters on the test set. The results prove to be similar, with a slight variation

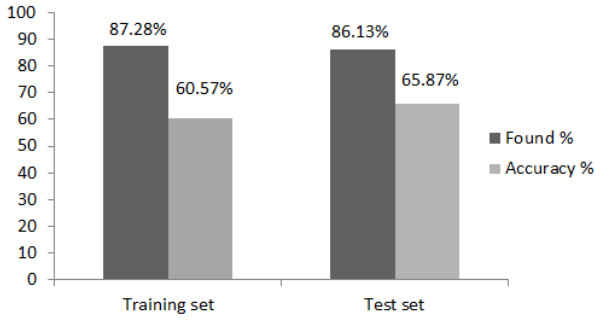

Figure 5. Comparing the results of all filters between the training and test sets

in both the number of results found and the accuracy of the results.

IX. Conclusion and Future Work

This paper set out to create a method for the selection of suitable mathematical translation candidates from queries received in Mxit lingo. Three techniques have been identified for this purpose and have been tested in combination to ensure the selection of the best method. It was determined that combining all three of the developed filters yields the most translatable results, it is however not the most accurate. If accuracy is a higher priority, then combining the number and symbol filters would be a better solution.

All of these techniques are light-weight, even using the combination of all three filters yield results in real-time. Future development could cross-reference the multi-filter processes to find a middle-ground between the number of results found and accuracy. However, this step might not be necessary as the speed at which the process takes place makes the time lost in processing false positives negligible.

As this study follows the design science research methodology [2], we have decided to use the design science guidelines [3] to evaluate whether the study's goals have been met. To address the first guideline, *design as an artifact*, the study has been structured to provide an implemented solution to detect translation candidates as it's final artifact.

The second guideline, *problem relevance*, is met by the fact that the study focuses on a real-world problem as demonstrated in the logs of Dr. Math. As it stands, the filters have been evaluated by comparing the results generated by the system itself. This does not take into account that the developed algorithms and testing procedures may be prejudiced or inaccurate. The third guideline, *design evaluation*, may be made more rigourous by using multiple coders and testing correspondence between coding through Krippendorff's alpha [21].

The study meets the fourth guideline, *research contributions*, by providing a means by which to enhance a real-world tutoring system applicable to over 30 000 school learners [1]. *Research rigour*, the fifth guideline, has been applied to the research by employing various methods on a training data set, learning from the first round of results and then applying changes before testing the methods on the training set again. Further rigour has been applied by testing the consistency of the methods and results on an alternate test data set.

The sixth guideline specifies that the *design should be a search process*. This search process was facilitated by employing three different filter types, a variety of natural language text processing techniques and finally modifying the techniques upon the receipt of initial data.

This paper serves to satisfy the last guideline, *communication of research*, which specifies that our research should be presented to an appropriate audience for verification.

The focus of this paper was on identifying techniques for the selection of suitable translation candidates. A future study may involve the development of rules for the translation of the selected candidates to mathematical equations.

References

[1] L. Butgereit and R. Botha, "A model to identify mathematics topics in MXit lingo to provide tutors quick access to supporting documentation." *Pythagoras*, vol. 32, no. 2, pp. 79–85, 2011.

[2] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research." *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, Winter 2008.

[3] A. R. Hevner, S. T. March, and J. Park, "Design science in information systems research." *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, March 2004.

[4] B. Goba, H. Morgan, K. Press, C. Smuts, and M. Van der Walt, *Study and Master Mathematics Grade 8*, 9th ed. Cambridge University Press, 2011.

[5] P. Carter, L. Dunne, H. Morgan, and C. Smuts, *Study and Master Mathematics Grade 9*, 12nd ed. Cambridge University Press, 2010.

[6] B. Goba and D. Van der Lith, *Study and Master Mathematics Grade 10*, 2nd ed. Cambridge University Press, 2008.

[7] D. Van der Lith, *Study and Master Mathematics Grade 11*, 5th ed. Cambridge University Press, 2008.

[8] ——, *Study and Master Mathematics Grade 12*, 9th ed. Cambridge University Press, 2010.

[9] South African Department of Education, "National Curriculum Statement Grades 10-12 (General) Mathematics," 2003.

[10] ——, "National Curriculum Statement Grades 10-12 (General) Mathematical Literacy," 2003.

[11] Various, "Webster's Unabridged Dictionary," Web, 2009, retrieved: 3 August 2012, http://www.gutenberg.org/ebooks/29765.

[12] W. Frakes and C. Fox, "Strength and similarity of affix removal stemming algorithms." *ACM SIGIR Forum*, pp. 26–30, 2003.

[13] J. Lovins, "Development of a Stemming Algorithm," *Mechanical Translation and Computational Linguistics*, vol. 11, no. 1 an 2, March and June 1968.

[14] M. F. Porter, *An algorithm for suffix stripping*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, pp. 313–316.

[15] C. D. Paice, "Another stemmer," *SIGIR Forum*, vol. 24, pp. 56–61, November 1990.

[16] Riptide Technology, "Easymark 2008 and EasyMark Worskpace 2008," Web, 2010, retrieved: 2 September 2012, http://www.riptidecc.com/softwaredevelopment.aspx.

[17] N. Schiller, J. Greenhall, J. Shelton, and A. Caramazza, "Serial order effects in spelling errors: Evidence from two dysgraphic patients," *Neurocase*, vol. 7, no. 1, pp. 1–14, June 2001.

[18] W. B. Cavnar and J. M. Trenkle, "N-Gram-Based Text Categorization," in *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, 1994, pp. 161–175.

[19] J. Mayfield and P. McNamee, "Single n-gram stemming," in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, ser. SIGIR '03. New York, NY, USA: ACM, 2003, pp. 415–416.

[20] Stapel, E, "Translating Word Problems: Keywords," Web, retrieved: 3 August 2012, http://www.purplemath.com/modules/translat.htm.

[21] A. F. Hayes and K. Krippendorff, “Answering the call for a standard reliability measure for coding data,” *Communication Methods and Measures*, vol. 1, no. 1, pp. 77–89, 2007.