SearchOnMath System Evaluation

José Carlos T. da Silva, jose.tobias@outlook.com, Federal University of Alfenas, *UNIFAL-MG*
Flavio B. Gonzaga, fbgonzaga@bcc.unifal-mg.edu.br, Federal University of Alfenas, *UNIFAL-MG*

***Abstract*—SearchOnMath (SOM) is a Mathematical Information Retrieval (MIR) system, indexing mathematical content from multiple Internet domains, using advanced techniques. Even though the SOM system is already complex, no evaluation has been done nor a method has been developed to assure its reliability and performance. This paper reports the evaluation of the SOM system, with the intent to give its engineers valuable information about the system's state, giving support to future development. A dataset of queries is built directly from the SOM's index, with the most computational complex formula from each group formed by a classifier. The system's responses for the dataset have their results categorized into expected and invalid, also having a zero-and-one inflated beta distribution fit. Such approach should allow SOM engineers to analyze the system results quantitative and qualitative aspects. From the $383\,130$ requests made to the SOM system, 99.94% of the responses returned with results categorized as expected, and only 66 responses returned containing no results. The distribution fit made possible to calculate values of expectancy and variance, allowing efficient comparison techniques to be applied over evaluations. We suggest that the work shown here can enhance system engineers' knowledge and improve the quality of the development process.**

***Index Terms*—information, retrieval, mathematical, system, evaluation, performance, assurance, classifier, clustering, analysis**

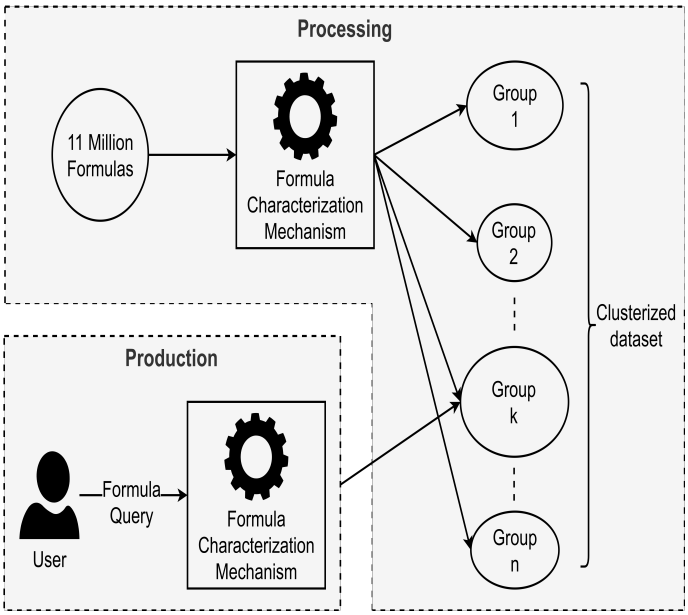

Figure 1. Division of SOM formulae index into groups based on their structure by a characterization mechanism .

I. Introduction

MATHEMATICAL Information Retrieval (MIR) is an Information Retrieval (IR) niche being mainly concerned with the representation, storage, organization and access of information items [1]. In the modern world, complex computer systems have been developed to execute tasks addressing those concerns, supporting human cognition on finding reduced groups of desired knowledge from large amounts of data [2]. Considering this scenario, the development of methods to assure that such systems are working as expected is crucial.

SearchOnMath[1] [3] (SOM) is a search engine developed with the purpose of supporting mathematical content search, where queries can be written combining text and/or mathematical expressions in LaTeX typesetting. Currently SOM indexes more than 11 million formulae from 7 domains[2], them being: Wolfram MathWorld, Wikipedia (English version), Stack Exchange Mathematics, Stack Exchange MathOverflow, Socratic, NIST DLMF and Planetmath. Because of the large size of its database, the SOM system implements a classifier and distribute the indexed formulae along multiple disjoint groups of similar formulae. Therefore, a formula submitted by a user has its structure checked, and the search is performed only inside the group where it is expected to have similar ones. Figure 1 illustrates this idea.

[1]https://www.searchonmath.com
[2]https://www.searchonmath.com/about

To evaluate this clustering process and search operation, our evaluation should be able to execute a search for every group and analyze the system's response. The SOM system has never been through this kind of evaluation, being expanded and updated based on the experience of its engineers. The problem of such approach to the development is that bugs introduced are only found late and new implemented techniques cannot always have their value stated precisely.

In this paper we present an evaluation over the SOM system, considering the clusterized search algorithm and operations executed over the formulae before and during the search engine process. The evaluation was designed with the purpose of verifying if the system is able to properly retrieve information from each of the groups. The paper is organized as follows. Section II explains how the dataset is built, presenting information to help interpret system's responses and the evaluation workflow. Section III brings the evaluation results and details about how the analysis is conducted to extract expected knowledge about the system's state. Directions about future work are given in Section IV. Conclusions are drawn in Section V.

II. Methodology

A. Dataset building

Due to its large amount of data, SOM is a distributed system that utilizes a clustering algorithm to execute its search

when a formula query is received. Consequently, to perform a wide and exhaustive system evaluation, we select the most complex formula for the system to process from each of the existing groups formed by the classification algorithm. The formulae selected build our dataset, that is a collection of queries to be submitted to the SOM system. Currently there are $414\,498$ groups in the SOM system clusterized index schema, where $383\,130$ of them contain formulae in LaTeX typesetting. Therefore, the built dataset consists of $383\,130$ formulae in LaTeX typesetting.

B. Evaluation Format

The evaluation consists of requesting each formula in the dataset to the system and analyzing its responses. Each formula from the dataset was requested to a SOM's REST API by an HTTP client. The API response is received as a JSON file, that should be parsed for analysis, containing an array with the top ten results. Each result is composed by the following data:

- **Formula**: the formula returned by SOM, considering the submitted query.
- **Similarity**: a value in the interval $[0.0, 1.0] \in \mathbf{R}$, where 1.0 means a perfect match between the submitted formula and the one that was found by SOM; and 0.0 the most different but still acceptable formula.
- **Language**: typesetting of the found formula, that should always be LaTeX.
- **Abstract**: page abstract where the formula was found.
- **Title**: page title.
- **URL**: page URL.

The similarity is the ranking criteria, such that the results are sorted inside the array in descending order. Based on this similarity parameter, the submitted formula can have its response classified as:

- **Expected**: first result containing similarity equals 1.0.
- **Invalid**:
  - **Not compatible**: first result does not have similarity equals 1.0.
  - **No results**: no results were returned for the submitted formula.

Since every formula from the dataset was obtained directly from the SOM's index, we expect that for each formula sent as a request the received results array response contains that formula as the top one with similarity equals 1.0, indicating an **expected** result. A result classified as **invalid** can happen if there is a bug in any of the phases, described as follows.

Internet domains indexed by SOM are composed by forums and other sites with similar collaborative content, based on users' contributions. Hence, it's very common to find non-LaTeX elements inside their formulae. An example of non-LaTeX elements which are usually found inside formulae are HTML elements. The problem is that MathJax[3] [4], a tool used by SearchOnMath to render LaTeX formulae, can't handle such HTML elements.

[3]https://www.mathjax.org/

Therefore, the SOM system must be able to receive LaTeX formulae, including "some errors" like HTML elements, find similar formulae and show results without rendering problems on MathJax. To assure that MathJax will render formulae on results page without errors, formulae contained into the database are submitted to two treatments described as:

- **HTML**: replace HTML elements for its respective LaTeX sequence.
- **textElements**: treats text input elements, deleting extra spaces, removing trailing commas or points, and other text format-oriented operations.

These treatments are usually sufficient to avoid errors during the rendering phase. However, textual elements that appear inside formulae (e.g. $b = (D - \text{root diameter})/2$) can't be identified as mathematical elements. In LaTeX typesetting there are multiple ways of to write fractions (e.g. '`{x}\over{y}`', '`\frac{x}{y}`' and '`\frac x y`' are all compiled to the same expression '$\frac{x}{y}$'), hence the database indexed formulae should be standardized. To meet such requirements, during the search phase another two treatments are made:

- **styleElements**: remove font styling elements such as '`\mathbb`' that turns 'R' into '$\mathbb{R}$' or '`\mathrm`' that turns 'R' into 'R'.
- **fractions**: standardize the multiple possible fraction notations to '`{x}/{y}`'.

We apply the *HTML* and *textElements* treatments to formulae before they are persisted into the SOM's database. A received query and a database selected formula go through the four defined treatments (*HTML*, *textElements*, *styleElements* and *fractions*) before a comparison method is called to assert their similarity. Figure 2 explains the treatment flow.

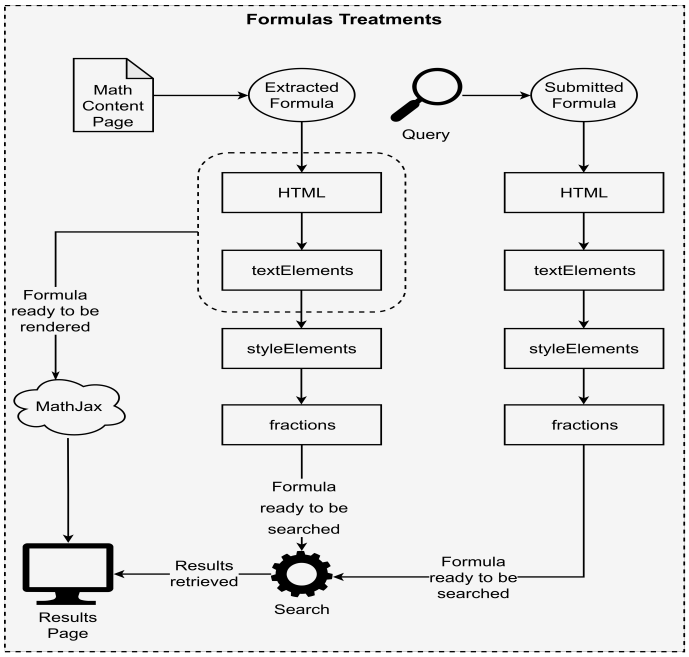

Figure 2. formulae treatments for SOM operations and appliance order.

We should now be able to understand why a search extracted from the system's own database, that doesn't return

the submitted formula as the top result with similarity equals 1.0, indicates a bug in one or more of the described formula treatment steps. That bug can lead the search to be executed inside a wrong cluster group or propagate to the similarity measurement algorithm. And by analyzing the request and response formulae, it is possible to identify which operation might contains a bug and move forward to fix its implementation.

C. Evaluation Process Workflow

Since the SOM system has never been through an evaluation, no workflow has been developed to support the procedure. Figure 3 gives us the activity pattern of the reported evaluation.

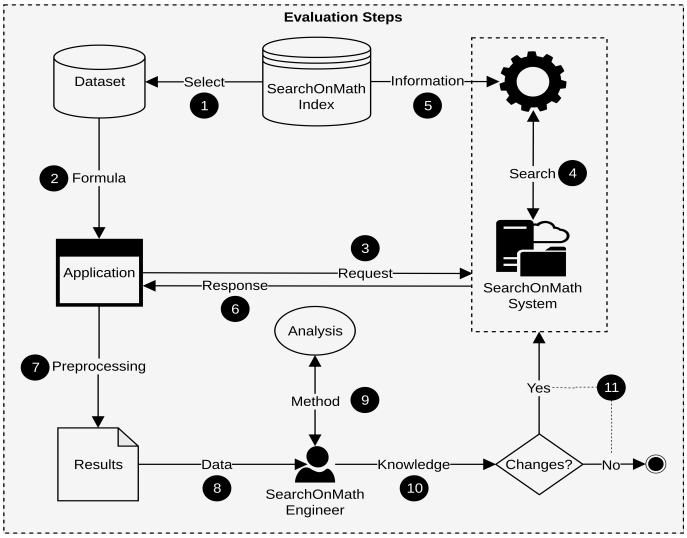

Figure 3. Step by step execution of an evaluation over the SearchOnMath system.

To begin we should **(1) select** the dataset formulae directly from the SOM's Index as explained in Section II-A, then the application can collect a **(2) formula** from the dataset and **(3) request** a query to the SOM's REST API. The system should then begin a **(4) search** phase, that will retrieve **(5) information** from its index and execute operations the sent query and formatting the results as explained in Section II-B. The **(6) response** is then sent back to the application, where a **(7) preprocessing** operation is applied allowing only important data to be stored for later usage. Such **(8) data** is passed to a SOM Engineer that should apply an analysis **(9) method** over it, giving them enough information to enhance their **(10) knowledge**. Therefore, that built knowledge might be used to answer questions correlated with the system, such as "are changes needed?". In case of a positive "**(11) yes**" answer engineers should now have enough support to apply changes to the system, otherwise with a negative "**(11) no**" answer the evaluation is considered done.

III. Results and Discussion

First we analyze how the data is categorized into expected and invalid, as set before in Section II-B. The Table I summarizes it, showing that we have roughly 0.06% of invalid responses, indicating a failure of system operations. There are two possible kinds of invalid results and their proportions are described in Table II. In this one we can see that we have only 66 requests where no information could be retrieved, and 165 cases where partial information was retrieved however not meeting the expectation for the first result.

Table I
Responses for the dataset formulae requests

Type	Quantity	%
Expected	382 899	99.94
Invalid	231	0.06
Total	383 130	100.00

Table II
Invalid responses

Type	Quantity	%
No results	66	28.57
Not compatible	165	71.43
Total	231	100.00

The categorization, based either on the first result and its similarity value leads us to another question, that is "how many submitted queries return one, two, three... until the total of 10 or more results?". The answer for this question is shown in the Figure 4. Given the complexity of the built dataset that is the highest possible for the system, Figure 4 shows that for 87.55% of the submitted queries only one result was returned. This shows that SOM also has an efficient algorithm to discard not relevant results.

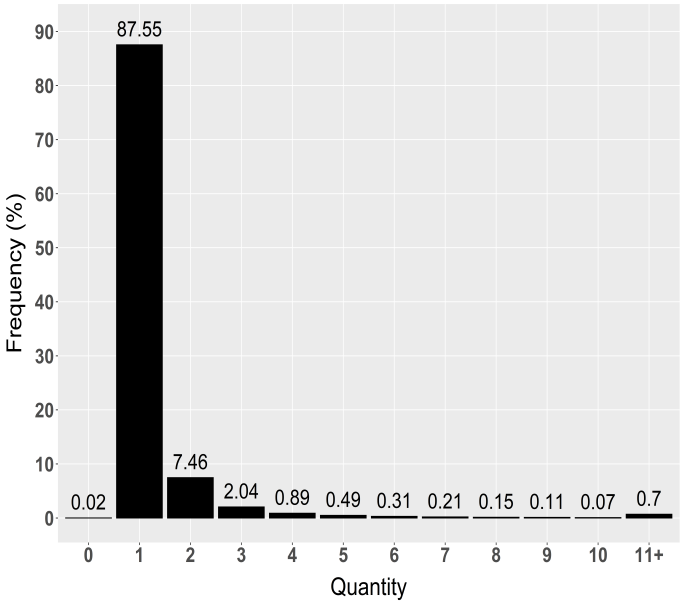

Figure 4. Frequency of results quantity by position for the dataset requests.

For a quality analysis of such results, the Figure 5 presents the similarity distribution by position for each of the 10 positions from the results array. The noticeable concentration of similarity values equal to 1.0 in the first results position is expected and happens because of the dataset complexity, being

correlated to the amount of responses classified as expected shown in the Table I. From the 2-nd to the 10-th positions, results present a similar characteristic being all inflated over the similarity values of zero and one, therefore we can fit a single distribution over them. The zero-and-one inflated beta distribution [5, 6], that mixes a Bernoulli distribution for values of zero and one with a beta distribution for the interval of $(0, 1)$, is chosen for this work.

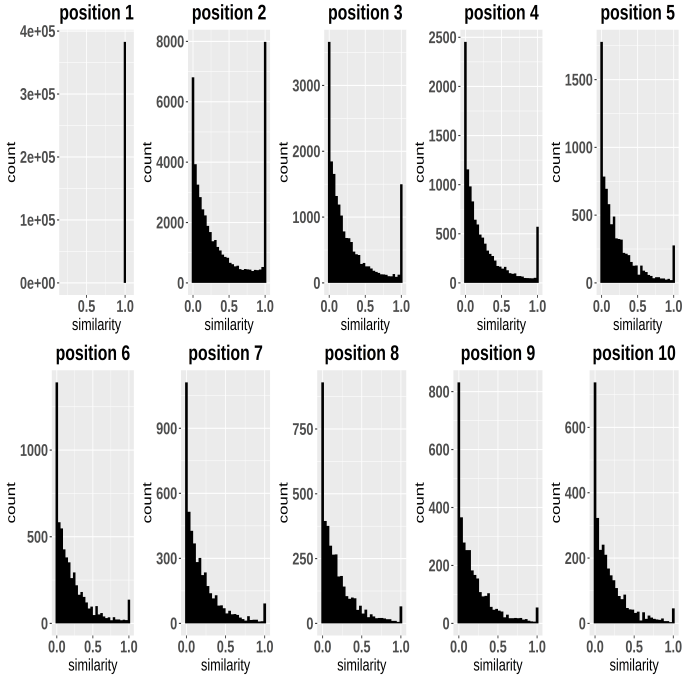

Figure 5. Frequency of results similarity by position. Notice the distribution similarity for positions from 2 to 10.

The beta distribution density function is defined with two parameters μ and ϕ. Such parameters were adjusted, from the 2-nd to the 10-th positions, by using the MLE (Maximum Likelihood Estimator) via Newton-Raphson method. Hence, allowing us to apply the expectancy and variance formulae as explained by Ospina and Ferrari [5]. The arithmetic mean and sample variance was used to the 1-st results' position due to its concentration over a single value. The results similarity expectancy ($E[X]$) by position and variance ($Var[X]$) by position can be seen in Table III.

Table III
EXPECTANCY OF SIMILARITY AND VARIANCE BY RESULTS ARRAY POSITION

Position	$E[X]$	$Var[X]$
1	0.9999641	0.0000097
2	0.6596060	0.1288510
3	0.3272972	0.1175128
4	0.2660084	0.0913476
5	0.2410593	0.0767783
6	0.2249611	0.0664863
7	0.2132837	0.0605189
8	0.2118640	0.0584650
9	0.2072782	0.0581457
10	0.2069176	0.0577592

As a way to improve the results quality comparison of evaluations, a graphical method is shown by Figure 6. The cumulative expectancy value is plotted as a curve, with error bars indicating the standard deviation of the similarity at given positions. The cumulative expectancy values shown for a given position $j \in [1, 10]$, are then obtained by summing up the expectancy values in the interval $[1, j]$. Both values, of expectancy and of standard deviation (which is the square root of the variance), were taken from the Table III.

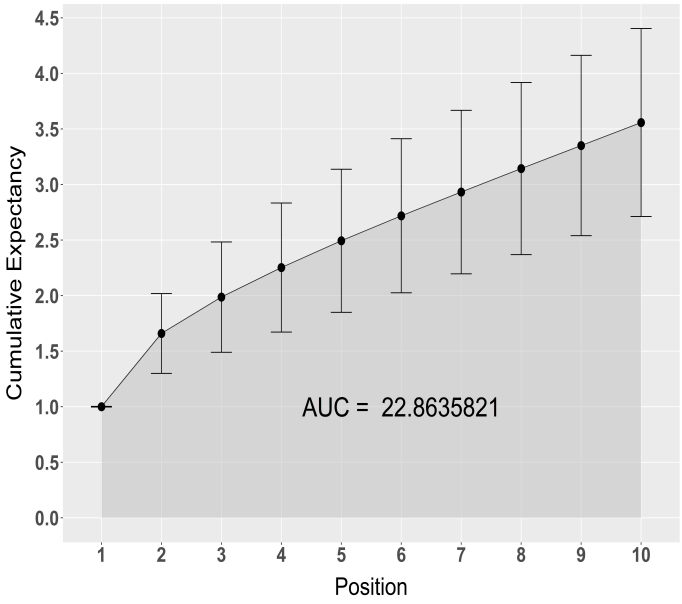

Figure 6. Cumulative similarity expectancy curve and standard deviation error bars by position. The AUC (Area Under Curve), of the cumulative similarity curve, is also displayed as an indicator of the responses' quality.

The AUC (Area Under Curve) was obtained by using the trapezoid method. The AUC acts as a quality indicator for the results obtained from the evaluation, and should be analyzed taking into consideration the standard deviation values and its effects towards the system.

IV. FUTURE WORK

Usually accepted metrics for performance comparison among systems, precision and recall [7] can also be obtained with the same created dataset or a part of it. The precision and recall metrics are taken by comparing the set A of expected results with the set B of results returned by the system. To obtain the set A, one should use the similarity calculation algorithm with every single dataset input against the whole database, and collect the the top $j \in [0, k]$ results ordered by similarity, where k is the size of the dataset. This process will not be utilizing the system's classifier algorithm for the search, the formulae treatment errors should be nullified by being applied on both sides equally and not propagated, and the specialist will be the algorithm used to measure the similarity parameter. To obtain the set B, one should request the same formulae from the dataset through the usual SOM system process and collect the top j results ordered by similarity.

V. Conclusions

The SOM system have never been through this kind of evaluation, therefore a method to support the procedure and analyze obtained data was developed and applied. Results obtained from the evaluation were categorized showing more than 99.96% of results as expected, while descriptive statistics showed that an efficient approach to discard not relevant results is also being taken. A zero-and-one inflated beta distribution was fitted to results by position, giving the possibility to analyze the quality of such responses. By being able to calculate the expectancy and variance of the data, is shown that a comparison of evaluations is possible given the observation of a graph with the expectancy by position cumulative curve and its AUC (Area Under Curve) value. Such evaluation methods should give SOM engineers a faster way to identify system inconsistent operations and to assert the value of new techniques to be implemented.

References

[1] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986. ISBN 0070544840.

[2] Justin Zobel. What we talk about when we talk about information retrieval. *SIGIR Forum*, 51(3):18–26, 2018. ISSN 0163-5840.

[3] Flavio Barbieri Gonzaga. SearchOnMath - Mathematical Information Retrieval System. https://www.searchonmath.com, Aug 2008.

[4] American Mathematical Society. MathJax - mathematical notation display for browsers. https://www.mathjax.org/, Jan 2009.

[5] Raydonal Ospina and Silvia L. P. Ferrari. Inflated beta distributions. *Statistical Papers*, 51(1):111, Mar 2008. ISSN 1613-9798. doi: 10.1007/s00362-008-0125-4. URL https://doi.org/10.1007/s00362-008-0125-4.

[6] Catherine Forbes, Merran Evans, Nicholas Hastings, and Brian Peacock. *Beta Distribution*, chapter 8, pages 55–61. John Wiley & Sons, Ltd, 2010. ISBN 9780470627242. doi: 10.1002/9780470627242.ch8. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470627242.ch8.

[7] Ellen M. Voorhees. The philosophy of information retrieval evaluation. In *Revised Papers from the Second Workshop of the Cross-Language Evaluation Forum on Evaluation of Cross-Language Information Retrieval Systems*, CLEF '01, pages 355–370, London, UK, UK, 2002. Springer-Verlag. ISBN 3-540-44042-9. URL http://dl.acm.org/citation.cfm?id=648264.753539.