Knowspace: Integration with Elastic Search

*Author: Amir S. Husain*

*Email: ah3548@nyu.edu*

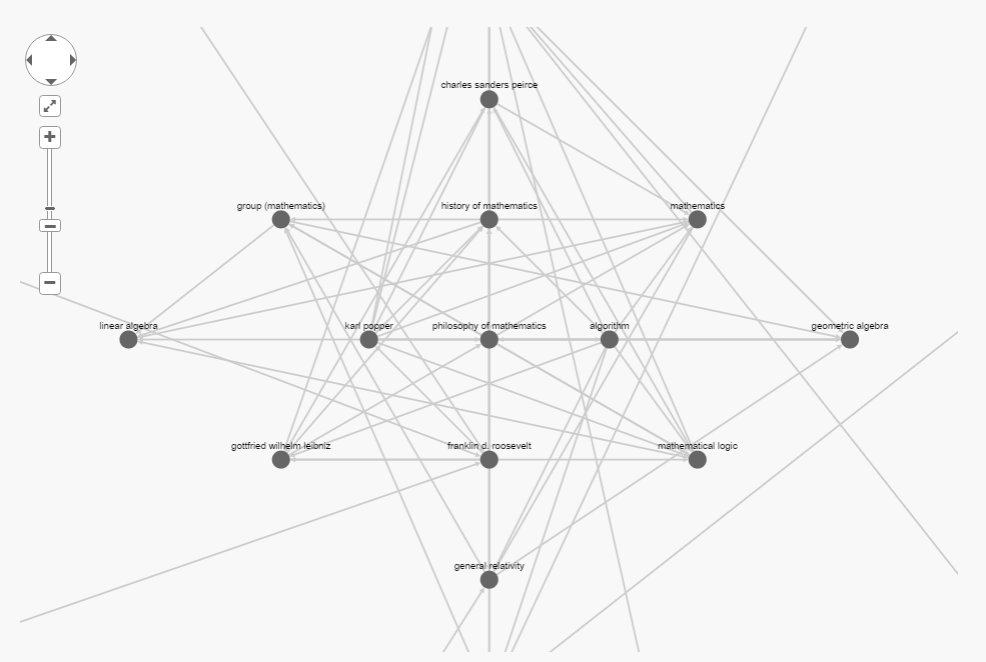
**ABSTRACT**

We aim to change the way adults (>14) approach education. Many learning styles exist and our approach is to let the user figure that out themselves. We do this by treating learning as a "Choose Your Own Adventure" game. By presenting the "Knowledge Space" through a node graph visualization we give the student a map. To present the associations and material we leverage Wikipedia to give clarification around complex topics.

**Keywords**

**1 INTRODUCTION**

The purpose of this project is to take an existing knowledge base (Linear Algebra) and present it in a new user-friendly manner. A node based graph approach is intuitive when discussing a subject but the methodology in generating this graph is the key piece of insight. Integrating with Elastic Search provided much benefit in its ability to store and index documents. The graph is generated off the ability to run a “more like this query” against a document. [1]



**2 SCOPE**

As a proof of concept we are going to present a use case to learn linear algebra using our tool. We are going to use elastic search to determine topic relationships. Elastic search will by default use tf-idf but this is configurable and can be run using other methods.

**3 MOTIVATION**

By trying to find documents that are highly related to each other we are trying to identify pre-requisites to knowledge. For example, if matrices are frequently mentioned in the document about linear algebra and/or the document about matrices frequently discusses linear algebra we can identify a potential dependence relationship. If we identify enough of these relationships we can find a “common core” to learn a subject (i.e. the node in figure 1 of history of mathematics) and potential fringe topics that may spark our interest (i.e. the node in figure 1 of franklin d. roosevelt).

**4. ARCHITECTURE AND TECHNOLOGIES**

**4 RESULTS AND COMPARISON TO EXISITING SOLUTIONS**

**4.0 Experiment Parameters**

When finding documents more like “Linear Algebra” we looked for the 7 most similar and for each result found the 7 most similar to that to generate the graph for a total depth of 3 include the root node. This is highly configurable and can be adjusted under different circumstances.

Some level of indexing is done pre user interaction and more data is indexed as user interact with the application. The below results are prior to any user interaction and ideally would further improve as more documents get indexed from intelligent user interaction.

**4.1 Standard Textbook**

|  |  |
| --- | --- |
| Introduction to Linear Algebra **[2]** *(<Chapter #> <Chapter Title #>)* | Wikipedia and Elastic Search*(<Distance from Linear Algebra Node> <Title>)* |
| **1 Introduction to Vectors** |  |
| **2 Solving Linear Equations** |  |
| **3 Vector Spaces and Subspaces** | **1 vector space** |
| **4 Orthogonality** |  |
| **5 Determinants** |  |
| **6 Eigenvalues and Eigenvectors** | **2 eigenvalues and eigenvectors** |
| **7 The Singular Value Decomposition (SVD)** | **1 singular value decomposition** |
| **8 Linear Transformations** |  |
| **9 Complex Vectors and Matrices** | **2 complex vectors and matrices** |
| **10 Applications** |  |
| **11 Numerical Linear Algebra** |  |
| **12 Linear Algebra in Probability & Statistics** |  |

**5 OBSTACLES**

There were some key obstacles in creating this tool that Elastic Search both introduced and/or solved.

One key issue I encountered was dealing with synonyms for articles. For example, “Coordinate” and “Coordinate System” both redirect to the same article and being able to store and index that in elastic search without duplication (as space was a concern) and re-fetching (as getting throttled was also a concern). The solution was to have a data point known as “akas” that stored an array of synonym. That way every time I found myself getting the same article returned, I would add that to the list of akas for future document retrieval.

Another issue I was previously encountering was that the Wiki API would return an html version of the page while this was not ideal for analysis. Through this project I came across a way to get a text only version that I stored alongside the html version. That way for analysis I used the text based version and for presentation in the UI I could use the html version.

One common issue I also encountered was my application was creating too many asynchronous requests to the Wiki API and separately through the Elastic Search API. This was overwhelming both my local Elastic Search instance and the remote Wiki API. Through some configuration of my node clients I found that I could limit each to a pool of 15 requests to not overwhelm either pipeline.

**6 FUTURE WORKS**

To address the shortcomings of creating multiple asynchronous requests I can find a way to bulk my read and write operations to the existing APIs. As for datasources, by integrating with other APIs such as (stack overflow, reddit, etc) I can grow the amount of material I can index and present.

Introducing this to a classroom setting where teachers or subject matters experts curate or publish their ideal graphs can further improve the results as nothing beats expert human curation.

To make the tool more engaging there is room for gamification that definitely can be explored.

**7 REFERENCES**

[1] Anon. More Like This Query | Elasticsearch Reference [5.3] | Elastic. Retrieved April 30, 2017 from <https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-mlt-query.html>

[2] Anon. Introduction to Linear Algebra, Fifth Edition (2016). Retrieved April 30, 2017 from <http://math.mit.edu/~gs/linearalgebra/>