An Information Retrieval Approach to Education Systems

Bhairav Mehta
University of Michigan
bhairavm@umich.edu

Adithya Ramanathan
University of Michigan
adithram@umich.edu

Abstract

This paper describes a system designed to enhance a students educational experience. By harnessing the resources available through online resources, combined with modern information retrieval techniques, we have built a novel system that gives the student access to a question-answer system to be utilized as a study aide. The question-answer system then provides the user with additional feedback such that specific weaknesses can be targeted and the student is informed of specific topics related to their weaknesses.

1 Introduction

Traditionally, education has always been a sequential process. Students learn algebra, geometry, and then calculus, and for some subjects, sequential learning offers a easy way to lay a natural foundation for future topics of study. Yet, in some cases, parallel learning of auxiliary yet related topics can help a student get through tough concepts. By providing a wider view of the topic in question, students gain more context, and can overcome mental roadblocks by bypassing them through temporarily focusing on other topics.

Further, education studies show that immediate feedback can help increase knowledge retention (Samuels and Wu). When compared to traditional delayed feedback methods such as grades on tests and quizzes, immediate feedback has been shown to improve understanding and comprehension. Both parallel learning and immediate feedback in education have empirically shown to increase the value of time spent in the classroom, but are idealistic by nature. Education is a one-size-fits-all system, and todays teachers often dont have the time to give each student the personalized attention needed to reinforce topics that the student is fundamentally weak on.

We develop the framework that takes initial steps in providing a solution to both the parallel learning and immediate feedback issues needed to increase the efficiency of our educational institutions. We present a system that extracts the key concepts from bodies of raw text and internet sources, and use these key phrases as a proxy for the knowledge a student hopes to gain from the particular group of sources. When presented with an answer, our system uses knowledge graphs to deduce what concepts the student is struggling with, and offers recommendations of other topics to study in parallel in order to improve knowledge retention of the main task. We stay true to the unofficial education-technology motto of minimizing teacher investment while maximizing student impact, and we conclude by offering future ideas on implementations and improvements of this system.

2 Related Work

Natural Language Understanding, a subset of Natural Language Processing, involves training computers to understand unstructured text. While Natural Language Understanding is considered an AI-Hard problem, a more accessible subset of this area is concerned with Information Extraction, or the task of retrieving structured information from unstructured data sources, such as raw text, images, or even video. Structured information, especially in a text-based setting, is often represented in the form entities and relationships.

Named Entity Recognition (NER) is a subset of information extraction and involves locating and classifying named entities in text into predefined categories such as the names of persons, organizations, locations, etc. Todays NER systems achieve near-human performance on popular datasets (Marsh and Perzanowski, 1998), but are often brittle and susceptible to the domain-transfer problem (Poibeau and Kosseim, 2001). This puts an onus on the researcher to provide the model with a labelled, relevant dataset, and currently, even with the state-of-the-art Conditional Random Fields models (Finkel et al., 2005), the domain transfer issue remains unsolved. In our work, we deal with examples for which we have related, labelled datasets. Therefore, while new work in neural NER and semi-supervised methods (Turian et al., 2010) (Lin and Wu, 2009) has shown promise in removing the prerequisite for a relevant, labeled dataset, we show promising results with more traditional models.

Relationship Extraction (RE) involves understanding how two entities are related. For example, a sentence John works at Google may translate into PERSON works for ORGANIZATION, with works for being the relation. Relationships are important in learning how much entities semantically mean to each other, and can stronger relationships can serve as a proxy for more importance of one entity to the other. Specifically, relationship extraction from raw text can be used to build knowledge graphs (Ramakrishnan et al., 2006). Knowledge graphs often hold facts as relationships, and in question-answering systems, can quickly infer answers by traversing relevant relationships between the entities in question. Knowledge graphs are important parts of other products like personal assistants, but to our knowledge, have not been directly applied to helping people in the traditional education process.

Education technology is defined as the ”the study and ethical practice of facilitating learning and improving performance by creating, using, and managing appropriate technological processes and resources” (Hlynka and Jacobsen). Education technology takes many forms, and while diverse in implementation, many educators agree that the ideal software solution will involve minimal teacher effort while still providing a substantial impact to students (Norris and Soloway).

3 Methodology

The system is constructed using five key steps. In order, the steps are as following: Raw Text Extraction, Named Entity Recognition, Topic Scoring / Ranking, Questioning / Answering, and Knowledge Graph Querying.

3.1 Datasets

Data was pulled and annotated from two main sources. Raw text was acquired by scraping Wikipedia using a seedlist of topics. Additionally, data used to train the Named Entity Recognition model was pulled using an annotated version of Groningen Meaning Bank.

3.1.1 Raw Text

Raw text is extracted by providing a seed list of URLs to a traditional, web-crawling spider, and the HTML on each page is processed and the raw text is extracted using the `BeautifulSoup` library. Our spider performs a depth-2 search from the seed URLs, meaning that it adds the links on the pages with the seed URLs, and then performs that process recursively on the next level as well. The assumption that we make is that a human expert provides these seed URLs by judging that the seed URLs are in fact relevant to the questions that will be inputted into the system at a later time.

3.1.2 Annotated Groningen Meaning Bank

The Groningen Meaning Bank(GMB) was developed at the University of Groningen (Bos et al., 2017). The bank itself features thousands of texts which have been tokenized, tagged for parts of speech and tagged for lexical categories. The specific dataset used in this project, found using Kaggle, is a further annotated version of the typical GMB and features labels for each term in the text in the form of named entity labels. Specifically, for this use case, terms are labelled using one of the following labels: geo (Geographical Entity), org (Organization), per (Person), gpe (Geopolitical Entity), tim (Time Indicator), art (Artifact), eve (Event), nat (Natural Phenomenon), O (Other). The dataset was tremendously useful as it provided a direct parallel to the output we wished to generate via our model. However, enhancements to the dataset do persist. Specifically, one area in which we may have benefitted from variation within the dataset is in the form of grouping of multi-term named entities. Currently, the system predicts upon raw text using data examples pulled from this bank, and each individual term is associated with named entity tag. This however, fails to consider that certain named entities are constructed of multiple terms. For example, the term Second Continental Congress is considered to be three unique named entities by this corpus rather than one large, multi term named entity.

3.2 System

3.2.1 Raw Text Extraction

Using the raw text harvested from the spider described in Section 3.1.1, we remove stopwords, and stem the remaining text using the NLTK Wordnet Lemmatizer (Fellbaum, 1998). With this preprocessed text, we concatenate all of the web pages to generate a tf-idf index (Ramos). As we assume that our text samples are representative of the true distribution of words, our tf-idf index serves as a proxy of how important and rare particular words are to the whole of the corpus. Through experimentation testing the ability of these indices to transfer across domains (i.e having been generated on a list of seed URLs from American History, how accurately could we gauge the importance of words from Psychology text), we find that these indices are also subject to the domain transfer issue discussed in the Related Work. To mitigate this issue, we envision a tf-idf index being generated per subject (i.e History, Psychology, etc).

3.2.2 Named Entity Recognition

The Named Entity module is utilized in two phases. First, there is a training phase where the model is trained to predict named entities in raw text using the Annotated Groningen Meaning Base as a training corpus. Secondly, there is a classification phase where the trained model is stored into a pickle file, and utilized to predict named entities from raw text.

Training begins with preprocessing and feature engineering of the training corpus. The corpus is preprocessed into sentence tuples. The entire training corpus is reconstructed as a list of tuples, where each sentence is broken down into tuples of word, part-of-speech, and its respective named entity tag. Given this reformatted training corpus, the data is converted into features extracting certain key identifiers for each word in a sentence such as case, position, part-of-speech, contextual part-of-speech, whether the word is a number, and whether the word is a title. Additionally, the features are engineered to consider both the previous and next word, and their respective values as features as well. At this point, the system has reformatted the Annotated Groningen Meaning Base into a collection of feature vectors.

For our use case, we utilize a Conditional Random Fields model. The CRF model performs a class of statistical modeling focused on conditional probability modeling. Additionally, CRF models fall into the class of sequential models, and is thus effective in this use case by considering neighboring terms and context when making conditional probability based predictions. The model is trained using the feature vectors previously constructed. The feature map appears as follows:

$$\phi(x_1, ..., x_m, s_1, ..., s_m)$$

where $x \in$ feature vectors and $s \in$ named entity labels.

The feature vectors are processed and considered using a log likelihood function:

$$L(w) = \sum_{i=1}^{n} log(p(s^i|x^i; w) - \frac{\lambda_2}{2}||w||_2^2 - \lambda_1||w||_1$$

Where the conditional probability function is modeled as:

$$p(s|x; w) = \frac{exp(w \cdot \phi(x, s))}{\sum_{s'} exp(w \cdot \phi(x, s'))}$$

And the parameter vector, where the two parameter terms (which helps alleviate complexity) are:

$$\frac{\lambda_2}{2}||w||_2^2$$

$$\lambda_1||w||_1$$

The parameter vector is estimated as:

$$w^* = \arg\max_{w \in R^d} L(w)$$

Based on training, and cross validation tests, we chose to modify the L1 Hyperparameter by increasing the value. This helped the model further focus on context rather than pure memorization and led to better empirical results. Once the model has been trained, we dump the model into a pickle file. The pickle file serves as a saved, binary export of the model such that we can later access the model to make predictions rather than retraining the model at each occurrence.

The second phase begins by taking in raw text as input. This raw text is then split into sentences, and each sentence is then further broken down into tuples. Using the NLTK (Natural Language Toolkit) library, the part-of-speech tagger is utilized to breakdown the sentence into tuples of term, and part of speech. At this point, we convert these reformatted sentences into feature vectors, in the exact same manner as the training data.

We then make predictions regarding each term.If the parameter vector is estimated, the most likely tag for a label can be found using.

$$s^* = argmax_s p(s|x; w^*)$$

At this point, we parse out all terms that are not considered named entities, and soley keep those original terms that are considered to be named entities.

3.2.3 Topic Scoring Ranking

Now, given a trained NER system and a tf-idf index that will serve as a proxy for importance, we can start to evaluate our system. During deployment, we envision that this is the first step an actual teacher would be involved.

The system prompts the user for a question title (which will be shown to the student), and then asks the user to provide source documents that the expert user deems to be relevant to the question posed. The source documents can be raw text, or URLs which are preprocessed the same way as described in Section 3.1.1. From there, we run each processed word through a scoring module

$$score_q(w) = tfidf(w) + a * SI(w) + b * NE(w)$$

where SI(w), short for semantic importance, is a multiplier if the particular word was in a section heading in the original HTML and a constant if from raw text, NE(w) is a multiplier if the word is present in any of the named entities extracted from the text using the NER model, and a and b are empirically defined hyperparameters. If the word comes from raw text inputted by the user, SI(w) $>$ 1, using the assumption that the user would have only inserted that text excerpt if it was actually relevant, whereas websites can contain lots of incorrect information. In addition, if the word is found inside of a named entity, we replace the word with the named entity itself. We then associate a thresholded list of {`word:  score`} pairs to the question, which serve as the key concepts associated with the question. In a deployed version of the system, we envision the sorted {`word:  score`} pairs to be returned to the user, and tweaked as necessary using expert knowledge.

3.2.4 Question Answer System

At this point, the system would be turned to the user, and the question inputted by the teacher would be posed to the student. In this preliminary work, we take a naive approach to the student scoring problem, awarding points to the student by the following criteria:

$$\text{score}_s(A) = \sum_{w_i \in A} \sum_{w_j \in E} [[w_i == w_j]] * score_q(w_j)$$

where A is a tokenized version of the students answer, and E is a tokenized version of the named entities generated from the teacher inputs the question.

This scoring methodology forces students to have exact answers, which is not ideal for educational software. While simple, future work could explore the use of synonyms, and words that have related meaning to those that come in as the student response for the use of scoring.

3.2.5 Knowledge Graph Querying

The knowledge graph is utilized in the project in an effort to take advantage of the concept of adjacent learning. As such, we utilize the constructed Google Knowledge Graph, and its corresponding API to access adjacent topics. Given that a student misses a certain concept when participating with the question answer system, these missed concepts are utilized as the terms with which we query the Google Knowledge Graph. We threshold the responses from the API request to the five most adjacent responses. These responses are then returned to the user as other suggested topics that would help the user better understand the material. Future implementations would utilize more specific corpuses to construct subject-specific knowledge graphs, for more relevant results to the user.

3.3 Examples

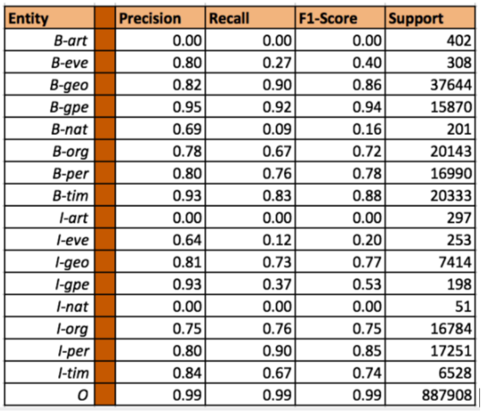
The use case we envision for our project is a student begins interaction with the system. At this point, the system asks the user a question such as Tell me about how World War II began?. The student then responds to this query by dictating all of the different pieces of information that they believe is relevant to answering this question. The system compares this information provided by the user against an internal scored, ranked list of terms relevant to the answer to the original question. Any terms that are ranked by the system, but missed by the student are then returned to the user as topics worth studying further. Additionally, the missed topics are input into a knowledge graph query such that topics related to the missed piece of information are also provided to the student.

In this way, our system targets specific weaknesses within a students domain knowledge of a subject. The system itself can be constructed for specific classroom curriculums, and provides an extension of the teacher to the student that allows the student to study more effectively when outside of the classroom itself.

4 Experiments

To evaluate results we performed 5-Folds Cross Validation to understand performance of our conditional random field model, utilized for named entity prediction. Additionally, to evaluate the system as a whole, which heavily depends on relevance feedback and a question answering system, we constructed an experiment in the form of a user study to understand the effectiveness of the system in serving as a study aid to students.

The 5 folds cross validation involves multiple rounds of random partitioning. Each round of partitioning involves separation into one training partition and one validation partition. The model is constructed using the training partition and tested using the validation partition. This process is repeated for each round and the results are averaged together. The randomness of the construction by design helps alleviate worries in the form of overfitting or underfitting. The model itself performed relatively well in cross validation testing. Certain key takeaways are still found however. Primarily, the data appears to be significantly dependent on the number of samples for each named entity type. Additionally, the system struggles with recall while performing relatively well in terms of precision.



Entity		Precision	Recall	F1-Score	Support
B-art		0.00	0.00	0.00	402
B-eve		0.80	0.27	0.40	308
B-geo		0.82	0.90	0.86	37644
B-gpe		0.95	0.92	0.94	15870
B-nat		0.69	0.09	0.16	201
B-org		0.78	0.67	0.72	20143
B-per		0.80	0.76	0.78	16990
B-tim		0.93	0.83	0.88	20333
I-art		0.00	0.00	0.00	297
I-eve		0.64	0.12	0.20	253
I-geo		0.81	0.73	0.77	7414
I-gpe		0.93	0.37	0.53	198
I-nat		0.00	0.00	0.00	51
I-org		0.75	0.76	0.75	16784
I-per		0.80	0.90	0.85	17251
I-tim		0.84	0.67	0.74	6528
O		0.99	0.99	0.99	887908

Figure 1: 5-Folds Cross Validation of NER

The user study was constructed to evaluate system performance on 5 example questions, and judged by 12 respondents. Each batch of 4 respondents was additionally averaged for an average evaluation score for the system. The user study resulted in a great variety of results, and has lead us to consider many further system enhancements in an effort to alleviate some of the concerns pointed out by this user study.

Questions Asked
Who was George Washington?
Name some important events in the Rev. War
How was life for the Iriquois Indians?
How did World War II Start?
What were some reasons for the Civil War?

Table 1: Questions Asked During User Study

In the user study, whose results can be seen in Appendix A, it can be seen that the system performs well when dealing with questions 4 and 5. However, when considering the ratings related to questions 1 and 3, the system has drastically worse performance. The results themselves are empirically obvious of the shortcomings. When looking at the related, suggested topics (as returned by the knowledge graph), a query for George Washington returns results related to completely unrelated topics to the question such as George Washington University. This completely unrelated response is representative of a key problem with the system - the knowledge graph has no specific contextual awareness. One potential solution to this problem would be an additional layer of post processing added to the knowledge graph module. Rather than relaying the first 5 responses from the knowledge query, the system should perhaps be capable of considering the returned responses and only relating those responses considered to actually be relevant.

5 Conclusion and Future Work

We present an alternative education solution through the use of modern information retrieval and machine learning techniques as described above. By harnessing the depth and breadth of resources available online, in combination with methodology that helps establish adjacent learning with immediate feedback, proven solutions to educational problems are being implemented. While the system as it stands provides the foundation for a framework focused on this education pipeline, glaring possibilities for improvement still exist. For example, related topics as related from the knowledge graph are not always intuitive or useful. Additionally, it can be argued that rather than providing the user with immediate feedback in the form of suggested adjacent topics, it may be more useful to use the adjacent topics to find another parallel question. In other words, rather than providing the user with the obvious response, position the user to continue their studying process in the form of a related question. This in turn would improve user engagement, and help the user organically relate topics themselves rather than being explicitly told of the relation.

References

Johan Bos, Valerio Basile, Kilian Evang, Noortje Venhuizen, and Johannes Bjerva. 2017. The groningen meaning bank. In Nancy Ide and James Pustejovsky, editors, *Handbook of Linguistic Annotation*, volume 2, pages 463–496. Springer.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. Language, Speech, and Communication. MIT Press, Cambridge, MA.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA. Association for Computational Linguistics.

Denis Hlynka and Michele Jacobsen. What is educational technology, anyway? a commentary on the new aect definition of the field.

Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 1030–1038, Stroudsburg, PA, USA. Association for Computational Linguistics.

E. Marsh and D. Perzanowski. 1998. Muc-7 evaluation of ie technology: Overview of results. In *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference Held in Fairfax, Virginia, April 29 - May 1, 1998.*

Cathie Norris and Elliot Soloway. The holy grail of ed tech apps: Require minimal teacher investment and provide maximal student impact.

Thierry Poibeau and Leila Kosseim. 2001. Proper name extraction from non-journalistic texts. In *In Computational Linguistics in the Netherlands*, pages 144–157.

Cartic Ramakrishnan, Krys J. Kochut, and Amit P. Sheth. 2006. A framework for schema-driven relationship discovery from unstructured text. In *Proceedings of the 5th International Conference on The Semantic Web*, ISWC'06, pages 583–596, Berlin, Heidelberg. Springer-Verlag.

Juan Ramos. Using tf-idf to determine word relevance in document queries.

S J Samuels and Yi-Chen Wu. The effects of immediate feedback on reading achievement.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394, Stroudsburg, PA, USA. Association for Computational Linguistics.

A Appendix: Results from User Study

In Table 2, we see the relevance ranking of the topics returned by the knowledge graph, when supplied the questions from Table 1. We then ask our users to rank the relevance of the top five returned topics, with 1 being the worst and 5 being the best. Some topics work better than others, due to the wide array of knowledge that Google's knowledge graph contains when compared to our controlled experiments. We believe that future work can help mitigate some of the low ranking responses by recommending relevant *questions* from the class that the current question is also in, rather than querying an outside API.

User	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**
1	1	3	2	4	4
2	2	3	2	5	4
3	1	4	2	3	3
4	1	3	2	4	4
5	2	5	2	5	4
6	1	3	2	5	5
7	2	3	1	4	4
8	1	3	2	4	4
9	1	4	3	5	4
10	1	3	2	4	3
11	2	3	2	5	4
12	1	4	2	5	4
Average	**1.33**	**3.42**	**2.00**	**4.42**	**3.92**

Table 2: Questions Asked During User Study