A Scalable, Flexible Augmentation of the Student Education Process

Anonymous MAIS Submission

Abstract

We present a novel intelligent tutoring system, which builds upon well-established hypotheses in educational psychology and incorporates them inside of a scalable software architecture. Specifically, we build upon the benefits of knowledge vocalization (Ausubel, 1961), parallel learning (Topping, 1996), and immediate feedback (Samuels and Wu, 2003) in the context of student learning. Driven by free, online resources, our work scales easily in terms of class size while still operating at the granularity of individual students. Additionally, we allow teachers to retain full control of the outputs, and provide student statistics to help them better steer their classroom discussions. Our experiments show promising results, and cement our hypothesis that the system is flexible enough to serve a wide variety of purposes in a classroom setting.

1 Introduction

One notable constant through the history of humanity is education. Despite changing mediums, from paper to pixels, the *structure* of education has stayed nearly identical: sequential, isolated learning, bookended by exams and benchmarked by grades.

In this paper, we develop a novel framework that can replicate some factors that are conducive to learning, as cited by educational psychologists; specifically, when students vocalize what they have learned, get feedback on those responses, and learn about related topics when they struggle with the main one, knowledge retention has been proven to increase (Topping, 1996). We ask students to vocally answer questions created by their teacher, who provided pertinent data sources when creating the question. We then perform semantic similarity tests between a student answer and the key concepts extracted from source data, and provide feedback to student about their performance. When struggling, we bring the student to other questions that will help cover their gaps in knowledge. We stay true to the unofficial education-technology motto of minimizing teacher investment while maximizing student impact (Norris and Soloway), and develop a pipeline that uses machine learning to augment the education process.

1.1 Related Work

Education technology (ET) is defined as the "practice of facilitating learning [by] creating, using, and managing appropriate technological processes" (Hlynka and Jacobsen). Many ET solutions do not make good use of either the educational infrastructure already in place, or the expertise of teachers in the classrooms. The leading ET applications offer deployment flexibility to the educator, constantly interact with the student, and offer ease of setup through pre-built curricula. Yet, many of these successful apps are still constrained by the fact that their materials are entirely expert-curated. Our approach builds upon the principles of interaction and feedback, but goes one step further. We offer educators a way to intelligently build curricula and provide recommendations to struggling students by using only a few, teacher-provided data sources.

2 Implementation

2.1 Pre-Teacher Setup

For each subject area supported by our framework, we create a tf-idf index (Ramos, 2003) using a handcrafted list of seed URLs that we believe are pertinent to the topic (i.e the subject of US History may have Wikilinks to different eras in American History). We use a web-spider to crawl the seed links to a configurable depth, and extract the text from each unique link as a separate document. We use standard text preprocessing techniques (Allahyari et al., 2017) in the creation of these indices, and mitigate the issue of domain transfer by maintaining multiple subject-based tf-idf indices. In parallel, we train a standard Paragraph Vector (Le and Mikolov, 2014) model with the GenSim Python Library (Řehůřek and Sojka, 2010) using the cleaned data, and store only the encoder for use in the recommendation engine described in Sec. 2.3.2.

2.2 Teacher Usage

Upon creating a class, the teacher selects a subject area that roughly corresponds to the class being taught, which, in the background, links a relevant tf-idf index to the class. The teacher, now creating questions, is asked to provide two pieces of information per question: a Question Title (seen by the student when asked to answer), and Data Sources. The data sources, links or blocks of text, are assumed to be relevant to the question and material covered in class, and are then preprocessed using the same standard techniques as in Sec. 2.1. Once we have cleaned text, we use a LSTM-CRF model (Lample et al., 2016), trained on the Annotated Groningen Meaning Base (Bos et al., 2017), to extract named entities (NE). In parallel, we run the text through TextRank (Mihalcea and Tarau, 2004; Barrios et al., 2016), and retrieve a list of key concepts (KC) from the source. We calculate the score for each word

using a weighted score of it's tf-idf weight, and indicator functions corresponding to its presence in either the NE or KC lists.

$$s(w) = TF(w) + \alpha I[KC(w)] + \beta I[NE(w)] \quad (1)$$

where $TF(w)$ is the tf-idf score of the word, and α and β are empirically-set hyperparameters. If appropriate, we combine the words into phrases using the two lists of NEs and KCs, as well as the NLTK Multiword Expression Tokenizer (Loper and Bird, 2002), and assign the phrase the sum of its component scores. We return to the teacher a list of `(concept, score)` pairs, who can then manually adjust any phrases and their corresponding score values. We also embed the raw text extracted from the question's associated data using the trained Paragraph Vector model described in Sec. 2.1, and store the "question embedding" in a database.

2.3 Student Usage

2.3.1 Answering

When a student enrolls in a class, he will see the questions proposed by the teacher, only with the question title. As the teacher picked data sources presumably related to class material, the student enables his microphone, and answers the question by speaking to the computer using classroom or background knowledge. To handle speech-to-text, we utilize Mozilla's open source implementation of DeepSpeech (Hannun et al., 2014; Mozilla, 2018), and with the transcripted text, we preprocess it with the same methods used in Sec. 2.1. From there, we tokenize the answer, and then score the answer by checking for token existence in the list of phrases associated with the question. We contribute the phrase's full score even for partial hits with student answers, and we acknowledge a more sophisticated scoring scheme could be used.

2.3.2 Recommendations

Along with the score and a visual representation of which words in his answer matched up with key concepts associated with the question, we also provide the student recommendations to other similar questions to enable parallel learning. We use compute the cosine similarity between the embeddings of the current question and all other questions created by the teacher within the particular class, and return the questions that are the three nearest neighbors in embedding space.

3 Results

For individual elements of the pipeline, such as the LSTM-CRF or the Paragraph Vector networks, we use many of the same training datasets and cross-validation procedures as described in the original papers.

We also conducted a user study to evaluate system performance (key concepts extracted, and recommended questions) on 10 example questions, and judged by 24 respondents (both teachers and students). We asked for 1-5 scale relevance ratings (5 being of high relevance) for each of the key concepts and recommended questions, and present truncated results in Table 1. We also gauge the importance and effectiveness of the three main embodiments of the effective learning hypothesis within our system: **(A)** Knowledge Vocalization, **(B)** Parallel Learning (via Recommendations), and **(C)** Immediate and Visual feedback, and present truncated results using a similar scale in Table 2.

Question	Q1	Q7	Q10
Avg. Relevance of KC	4.33	3.50	4.00
Avg. Relevance of Rec. ?s	1.33	3.20	4.50

Table 1: Sampled relevance scores, avg'd

	(A)	(B)	(C)
Component Importance	3.75	4.33	2.50
Component Effectiveness	4.25	2.33	2.33

Table 2: Importance and Effectiveness Scores, avg'd

Our most surprising conclusions came from a pilot program conducted in *Anonymized Area* Schools, after the a web-based implementation of the system had been in use inside of 3 classrooms studying various Advanced Placement (AP) subjects. All three teachers, given just the system, were using it in completely different ways: one as a total homework substitute, one as a homework add-on, and another as a independent (i.e no teacher enforcement of usage) self-study tool for annual AP exams. These results show that even a barebones implementation of this framework, one driven by open-source information sources and widely available machine learning algorithms, can be easily molded to varying use cases with no extra effort from educators.

In addition, discussions with the pilot program's teachers showed that the system's class performance statistics were overwhelmingly described as the most important features. Class-level feedback allowed for more focused discussions during classtime with students. During the short pilot test, we were told that teachers' day-to-day schedules became more dynamic, and time devoted to discussing each topic was driven by the performance of the students, rather than just blindly assigned from the teacher's intuition.

4 Conclusion

We present a simple yet effective system that builds on popular hypotheses in educational psychology, and reproduce them in a scalable software framework. We provide a flexible solution that can serve a wide variety of purposes in a classroom, while keeping educators in the loop every step of the way. We present empirical evidence that cements our original hypothesis about key factors for effective learning, and our work shows that the right mix of machine learning models can provide students and teachers enormous impact when turned towards an education setting.

References

Mehdi Allahyari, Seyed Amin Pouriyeh, Mehdi Assefi, Saied Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut. 2017. A brief survey of text mining: Classification, clustering and extraction techniques. *CoRR*, abs/1707.02919.

David P. Ausubel. 1961. In defense of verbal learning. *Educational Theory*, 11(1):15–25.

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *CoRR*, abs/1602.03606.

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.

Awni Y. Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. 2014. Deep speech: Scaling up end-to-end speech recognition. *CoRR*, abs/1412.5567.

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

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *CoRR*, abs/1603.01360.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. *CoRR*, abs/1405.4053.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*.

R. Mihalcea and P. Tarau. 2004. TextRank: Bringing order into texts. In *Proceedings of EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing*.

Mozilla. 2018. mozilla/deepspeech. `https://github.com/mozilla/DeepSpeech`.

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

Juan Ramos. 2003. Using tf-idf to determine word relevance in document queries.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. `http://is.muni.cz/publication/884893/en`.

S J Samuels and Yi-Chen Wu. 2003. The effects of immediate feedback on reading achievement.

K. J. Topping. 1996. The effectiveness of peer tutoring in further and higher education: A typology and review of the literature. *Higher Education*, 32(3):321–345.