Maximizing student benefit from instructor interactions with MOOC discussion forums

Glenn M. Davis
Stanford University
450 Serra Mall
Stanford, CA
gmdavis@stanford.edu

Yanyan Tong
Stanford University
450 Serra Mall
Stanford, CA
yanyan.tong@stanford.edu

Aymen Riwzwan
Stanford University
450 Serra Mall
Stanford, CA
aymen@stanford.edu

Andreas Paepcke
Stanford University
450 Serra Mall
Stanford, CA
paepcke@cs.stanford.edu

ABSTRACT

We developed and tested algorithms to accurately identify MOOC discussion forum threads that would benefit from immediate instructor intervention. An automatic classifier was first used to generate sentiment, confusion, urgency, and question/answer/opinion status at the post level, and discussion forum threads were then reconstructed and ranked using algorithms that make use of these generated tags. We compared our four treatment algorithms (*urgency*, *confusion*, *last_answers*, and *question_votes*) with the three control algorithms currently used by edX to sort forum threads (*timestamp*, *first_votes*, and *num_posts*), using a corpus that includes all 1,182 threads from three iterations of an archived Statistics in Medicine MOOC hosted by Stanford on the Open edX platform. Nine domain experts in statistics rated the top three threads ranked by each algorithm for potential contributions to student learning and degree of urgency. Two of our treatment algorithms (*confusion*, *question_votes*) significantly outperformed the control algorithms at identifying urgent threads that would contribute to student learning.

Keywords

Discussion forums; massive open online courses (MOOCs); natural language processing; sentiment analysis

1. INTRODUCTION

MOOCs (massive open online courses) allow instructors to present course material to class sizes numbering in the tens of thousands or more, with little additional investment of time on the part of the instructor required to support increasingly larger classes. Although lecture videos and course materials easily scale to accommodate any number of users, the discussion forum component of many MOOCs does not scale, and an instructor who wishes to answer all student questions will be quickly overwhelmed in large classes.

Figure 1 displays how discussion forum threads are currently presented to both instructors and students on the edX platform.[1] Even when the discussion forum is separated into subforums for different topics (e.g., Week 1 questions, Week 2 questions, Administrative), there may be hundreds of threads in a single subforum. A short preview of the first post is presented under the title, but there is no indication of how the thread may have progressed through its lifespan. Thus, although a problem presented in the first post of a thread may have been answered by another user later in the thread, there is no way to know that a student question has been resolved without opening a thread and reading through it. As such, instructors have no easy method of determining which threads on the discussion forum still contain urgent questions or student misconceptions that require instructor attention without performing a manual scan of each thread.

Previous research has used natural language processing (NLP) tools to begin to analyze MOOC discussion forums in a more scalable manner. Agrawal et al. [3] developed a system that automatically detected whether forum posts expressed confusion, and recommended segments of course videos with related keywords to the confused users. Building on this work, Wei et al. [5] were able to detect confusion, urgency, and positive or negative sentiment based on forum posts in several different domains, including humanities/sciences, medicine, and education. Further researchers have created models to identify whether discussion forum threads were content-related or unrelated [6], and to group forum threads together by topic and subtopic [4].

However, instructor capacity to interact with students on discussion forums does not scale, and thus the workload required to provide useful answers to student questions currently grows near-linearly as the class size increases. Our work develops and tests algorithms to identify the threads

[1]This screenshot was taken from the archived MIT course Introduction to Probability - The Science of Uncertainty.

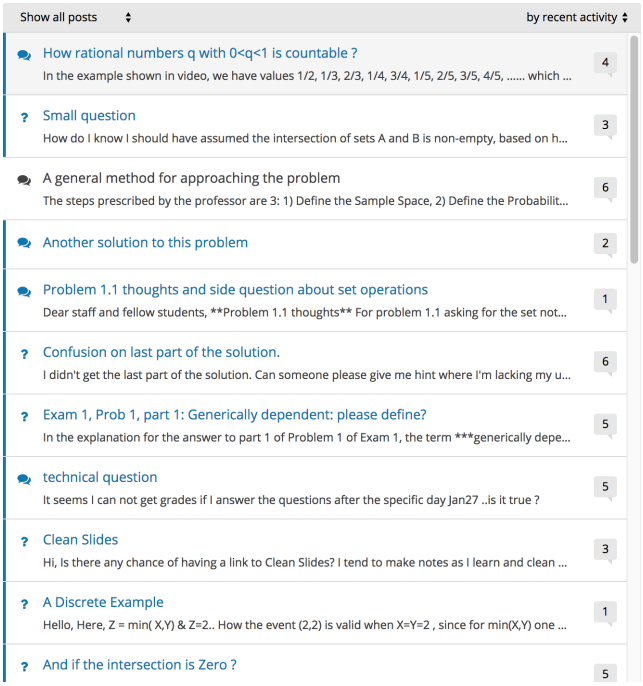

Figure 1: Current presentation of discussion forum threads on a busy edX MOOC.

from a large corpus that would most benefit from instructor intervention.

We develop multiple algorithms, each of which incorporates both post-level and thread-level factors. Previous work has mainly focused on only one of these levels: we combine information from a natural language classifier that identifies relevant posts with the surrounding context of the thread in which the post appears in order to estimate whether the post is still relevant by the current end of the thread.

A system that implements such algorithms would improve the scalability of instructor interaction with discussion forums, as instructors could always attend to the highest priority threads, regardless of total forum size or number of less relevant threads. Figure 2 shows a mockup of a plug-in to the popular Open edX platform that classifies and sorts discussion forum threads using our new algorithms. This proposed system would allow instructors to more quickly identify threads that would benefit from their intervention, reducing the workload required to interact effectively with students on the discussion forum. We present the technical solution needed to realize such a system.

2. METHOD

2.1 Dataset

Our dataset is comprised of all 1,182 discussion forum threads (4,899 posts) from three iterations of a Stanford Statistics in Medicine edX MOOC. This data was obtained from Stanford Datastage[1], and spanned three offerings of the course over the years 2014 and 2015.

2.2 Classifier

We used a natural language classifier to obtain information about the potential relevance of each post in the dataset. This classifier was originally developed by Agrawal et al. [3] to detect confusion in discussion forum posts, and incorporates multiple classifiers that each tag every post with the binary characteristics sentiment (positive or negative), urgency, confusion, question, answer, and opinion.

We used the Stanford MOOCPosts Dataset as a training set, mirroring the approach used by Agrawal et al. [2] The Stanford MOOCPosts Dataset contains 29,604 posts taken from 11 Stanford University edX courses, and includes 2013 and 2014 offerings of the same Statistics in Medicine course used in our test set. The 2014 offering, which was included in both the training set and our test set, contains 1,218 posts. Human coders rated each post for positive or negative sentiment, urgency, and confusion level (1-7 scale), as well as whether the post was a question, answer, or opinion (binary).

The classifier takes these human ratings and uses a bag-of-words approach to develop associations between these six features (sentiment, urgency, confusion, question, answer, opinion) and the words present in each post. It then produces binary tags that represent presence or absence of each of the six features for all posts in the test set; here our 4,899-post Statistics in Medicine dataset.

2.3 Other information

Our algorithms also make use of other post-level information available in the dataset. This information includes the number of upvotes and downvotes received for the post, the timestamp of the post, and the user ID of the post creator.

We reconstructed the 1,182 threads in their original chronological sequence of posts, using the thread IDs and timestamps for each post. This allowed us to develop algorithms that made use of post-level information at different locations in the thread, such as the beginning and end. Further, this handling of discussion forum data at the thread level more closely resembles the current paradigm of instructor interaction with the forums. As shown in Figure 1, users are initially presented with the different threads and only a snippet of the first post in each thread, and must select a thread before being able to read the individual posts.

2.4 Algorithms

We first implemented three control algorithms that mimic the default sort options available on edX discussion forums at the time of writing this paper.

First_votes replicates the behavior of the edX sort "by most votes" option, ranking threads by number of upvotes for the first post in the thread. Upvotes for all other posts in the thread are ignored.

Num_posts replicates the edX sort "by most activity". Threads are ranked by the total number of posts in the thread, with more active threads (more posts) ranked higher.

Timestamp replicates the edX sort "by recent activity". Threads are ranked by the timestamp of the latest post in the thread,

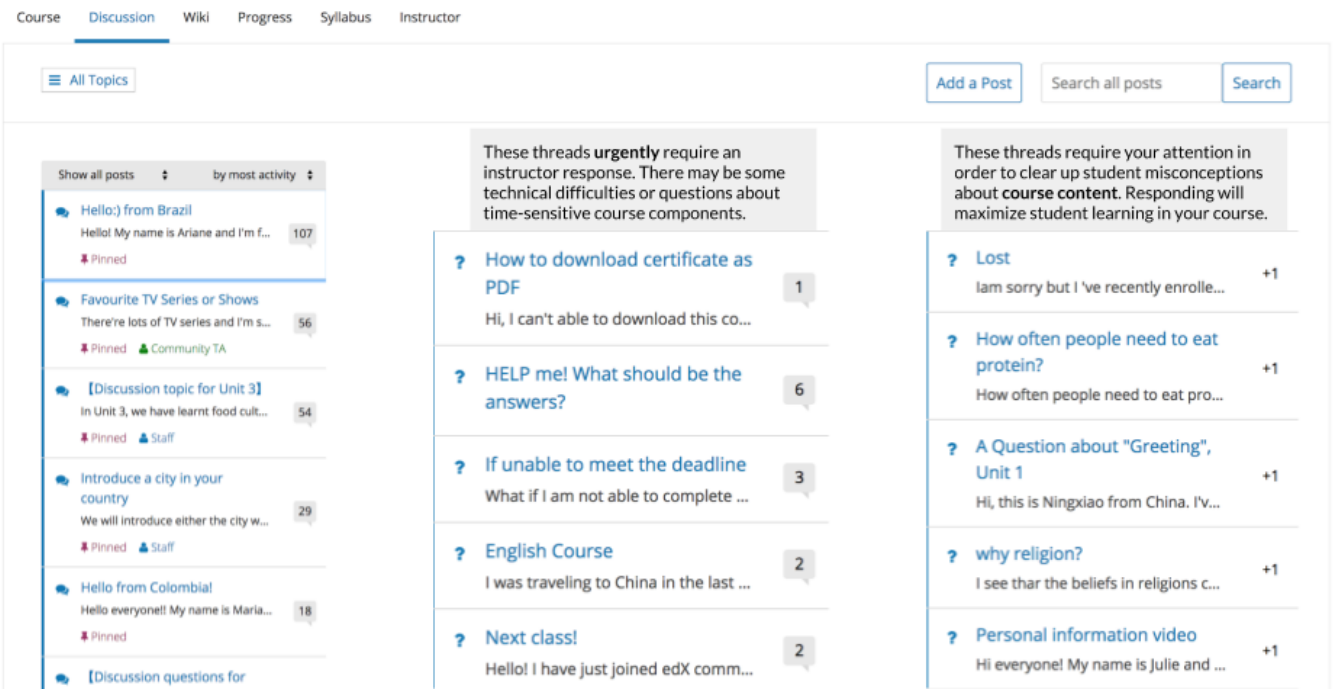

Figure 2: Mockup of an instructor interface implementing these algorithms.

with later (more recent) posts ranked higher. Timestamps of all posts in the thread other than the latest are ignored.

We then developed four treatment algorithms that incorporate the post-level information available in the original dataset, the post-level tags created by our classifier, and thread-level information.

Urgency uses the binary *urgency* tags created by the classifier. Threads are ranked by the proportion of posts in the thread classified as urgent. As the corpus contained several threads that consisted of a single post tagged as urgent, we set a minimum thread size of five posts.

Confusion uses the binary *confusion* tags created by the classifier. Threads are ranked by the proportion of posts in the thread classified as confused. Again, we set a minimum thread size of five posts.

Last_answers traverses the thread in chronological order and finds the most recent post tagged as a question by the classifier. The thread's score is determined by the number of posts tagged as answers by the classifier that follow (i.e., are more recent than) this last question. Threads are inversely ranked by the number of answers after the final question in the thread, based on the intuition that the issue presented in the question is more likely to have been adequately resolved as the number of following answers increases. Using this algorithm, threads that have no answers following the final question are ranked highest.

Question_votes uses the post-level upvotes data available in the original dataset, but only considers upvotes for posts tagged as questions by the classifier. The total number of upvotes on question posts is divided by the total number of question posts to produce the average number of upvotes for questions in the thread, and threads are ranked by this average.

We thus used these seven algorithms to each rank all 1,182 threads in the dataset:

Treatment algorithms:
Urgency: Proportion of posts in the thread classified as urgent, minimum five posts.
Confusion: Proportion of posts in the thread classified as confused, minimum five posts.
Last_answers: Number of posts classified as answer since the last question in the thread (fewer = higher rank).
Question_votes: Average number of upvotes for questions in the thread.

Control algorithms:
First_votes: Number of upvotes for the first post in the thread.
Num_posts: Total number of posts in the thread.
Timestamp: Timestamp of latest post in the thread (later = higher rank).

3. HUMAN RATER STUDY

We verified the accuracy and performance of our algorithms through a human rater study with nine domain experts in statistics. All of the domain experts were graduate students in the statistics department and/or had experience teaching statistics courses.

3.1 Procedure

The top three threads ranked by each of the seven algorithms were extracted from the dataset. However, one thread was

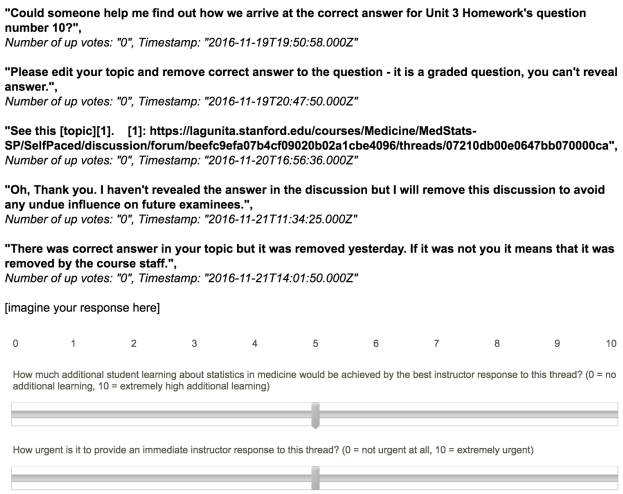

Figure 3: Human rater study interface.

ranked in the top three by both *first_votes* and *num_posts*, and the duplicate was removed. This resulted in a total of 20 unique threads that were used in the human rater study.

A Qualtrics survey[2] was used to present all 20 threads in a randomized order to each participant. For each thread, the domain expert was presented with all of the posts in chronological order and asked to imagine him/herself in the role of a course instructor deciding whether or not to add a post to the thread. If there were more than 20 posts in the full thread, the first 5 posts and the last 15 posts were presented with the text "[*(number)* posts omitted here]" between them.

For each thread, the experts rated how much additional learning would be brought about by adding an instructor response to the thread, as well as how urgently the thread required instructor attention, on 0-10 scales. Figure 3 shows the survey interface used by the domain experts. Experts were paid $15 for their participation in the survey, which took approximately 20-40 minutes to complete.

3.2 Results

The mean scores for the domain expert ratings of the top three threads ranked by each algorithm are shown in Table 1.

Figures 4 and 5 display the domain experts ratings in more detail. Each box plot represents a total of 27 ratings for the threads corresponding to that algorithm (9 domain experts, each of whom rated the top 3 threads ranked by that algorithm).

The box plots show that the group of treatment algorithms generally outperformed the control algorithms on both perceived marginal learning benefit from instructor response (Figure 4) and perceived urgency of instructor response (Figure 5). The position of each box plot along the y-axis can be interpreted as the degree to which the domain experts

[2]www.qualtrics.com

Table 1: Mean scores provided by domain expert raters for the top three threads ranked by each algorithm. Treatment algorithms are above the divider, control algorithms below.

	Student learning	Urgency
urgency	3.85	4.59
confusion	5.11	6.37
last_answers	3.96	4.22
question_votes	5.67	5.78
first_votes	1.85	1.30
num_posts	2.93	3.04
timestamp	3.30	4.04

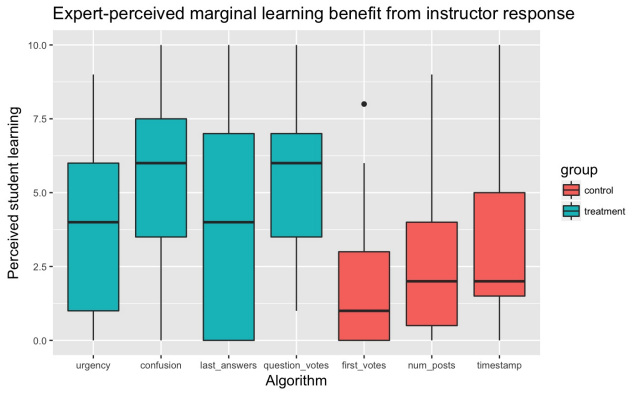

Figure 4: Domain expert ratings of perceived student learning for the top 3 threads ranked by each algorithm.

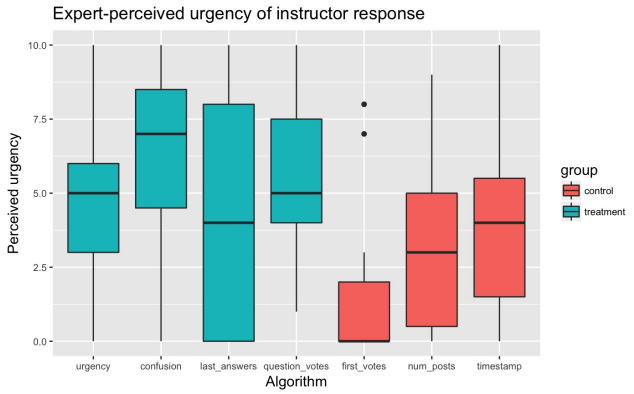

Figure 5: Domain expert ratings of perceived urgency for the top 3 threads ranked by each algorithm.

agreed that the algorithm had selected threads that would benefit from instructor response (Figure 4) and were urgent (Figure 5).

Notably, the control algorithm *first_votes* performed very poorly on both metrics, suggesting that the edX default sort "by most votes" option is not effective at identifying threads that require instructor intervention. On the other hand, the treatment algorithms *confusion* and *question_votes* were perceived as the most effective algorithms for both metrics.

3.2.1 Student learning

A one-way within-subjects ANOVAs indicated that the algorithms differed in their ability to select threads that would bring about additional learning, $F(6, 48) = 10.05$, $p < .001$. A Tukey post-hoc test revealed that the group of treatment algorithms outperformed the group of control algorithms, $p < .001$.

Additional Tukey post-hoc tests were conducted to compare each treatment algorithm with the best-performing control algorithm, *timestamp*. Results showed that *question_votes* significantly outperformed *timestamp*, $p < .05$, but *confusion*, *last_answers*, and *urgency* did not, ps $> .05$. *Question_votes* also outperformed the other control algorithms (*first_votes*, *num_posts*), ps $< .01$.

3.2.2 Urgency

A one-way within-subjects ANOVAs indicated that the algorithms differed in their ability to select urgent threads, $F(6, 48) = 12.13$, $p < .001$. A Tukey post-hoc test revealed that the group of treatment algorithms outperformed the group of control algorithms, $p < .001$.

Additional Tukey post-hoc tests were conducted to compare each treatment algorithm with the best-performing control algorithm, *timestamp*. Results showed that *confusion* significantly outperformed *timestamp*, $p < .05$, but *question_votes*, *last_answers*, and *urgency* did not, ps $> .05$. *Confusion* also outperformed the other control algorithms (*first_votes*, *num_posts*), ps $< .001$.

4. CONCLUSIONS

As a group, the treatment algorithms significantly outperformed edX's default sorting algorithms on expert-rated perceptions of additional student learning that would be achieved by an instructor response, and urgency of providing an instructor response.

Individually, the treatment algorithm *confusion* (proportion of posts tagged as confused in the thread) outperformed the best control algorithm *timestamp*, at identifying threads that would maximize student learning with an instructor response. The treatment algorithm *question_votes* outperformed *timestamp* at identifying threads that urgently required an instructor response.

These results establish that it is possible to generate algorithms to sort MOOC discussion forum threads in ways more beneficial to instructors than the default sort options provided on the edX platform. Two of our treatment algorithms, *confusion* and *question_votes*, were each rated as superior to all currently existing sort methods on the platform for one of the two metrics.

Implementing a system that uses these algorithms to sort discussion forum threads, as proposed in Figure 2, would allow for instructors to more quickly find threads that would benefit from intervention, and thus maximize student benefits from instructor time spent interacting with the forum.

4.1 Future work

Future work on this project aims to develop a classifier that uses non-binary ranges for characteristics. This increased fidelity will allow for more nuanced algorithms to be developed that take into account small-scale changes in sentiment, confusion, and urgency level from post to post.

This project examined the effectiveness of treatment algorithms in identifying threads in an archived discussion forum. Implementing such a system in a live MOOC would allow for longitudinal studies of the effects on student learning achieved by more efficient instructor interactions with the forum.

5. REFERENCES

[1] CAROL learner data documentation—Stanford MOOC Dataset. `http://datastage.stanford.edu`. Accessed: 2018-03-03.

[2] The Stanford MOOCPosts data set. `http://datastage.stanford.edu/StanfordMoocPosts/`. Accessed: 2018-03-03.

[3] A. Agrawal, J. Venkatraman, S. Leonard, and A. Paepcke. YouEDU: Addressing confusion in MOOC discussion forums by recommending instructional video clips. *Proceedings of the 8th International Conference on Educational Data Mining*, pages 297–304, 2015.

[4] J. M. Vytasek, A. F. Wise, and S. Woloshen. Topic models to support instructors in MOOC forums. *Proceedings of the Seventh International Conference on Learning Analytics & Knowledge - LAK '17*, pages 610–611, 2017.

[5] X. Wei, H. Lin, L. Yang, and Y. Yu. A convolution-LSTM-based deep neural network for cross-domain MOOC forum post classification. *Information*, 8(3):92, jul 2017.

[6] A. F. Wise, Y. Cui, and J. Vytasek. Bringing order to chaos in MOOC discussion forums with content-related thread identification. In *Proceedings of the Sixth International Conference on Learning Analytics & Knowledge - LAK '16*, pages 188–197, 2016.