Automated Comment Scoring

Tom Yedwab, Kyle Hamilton, Cory Kind
December 2016

Abstract

The widespread use of online platforms to deliver educational content has created opportunities for the platforms themselves to actively curate content for students, potentially improving outcomes. This research applied a variety of modern techniques (LSTM, Multilayer Perceptron Neural Networks, and CNNs) to automatically identify high-quality comments on Khan Academy, "the largest school in the world"[1]. Results indicate a marginal accuracy improvement over the human baseline and discuss some of the key challenges other researchers must address when tackling this problem. Findings from this research have implications for creators and administrators of educational technology, as well as for instructors in these courses.

Keywords: Automated comment scoring systems; assessment; educational technology

Introduction

Our goal for this project was to build a model that automatically identifies high-quality comments among those submitted by users on a given topic. There are several different types of educational platforms where we can see this technology being used. In settings where there is a formal instructor and students receive evaluations on their comments, these posts can be used for both formative assessment (to gauge student progress and facilitate learning) and summative assessment (final scoring)[2]. The application of NLP techniques to reliably and efficiently score discussion post data can help reduce the burden on the instructor. This would allow them to focus more time and attention on higher-value activities. Similar research is also being done to help instructors build a comprehensive view of course submissions more easily.[3]

On platforms like Khan Academy where the "instructor" role is played by other students and in part by the platform itself, the ability to automatically differentiate high-quality comments from low-quality comments would provide a unique competitive advantage. Khan Academy is one such platform, but there are many others as well (e.g., Coursera, EdX). Particularly in the case where students have no expectation of direct interaction with an instructor, automatic comment scoring can enable the platform to better fill that void and provide a more curated experience for the learner– the algorithm can filter through a large number of comments and bring to the top those that learners are going to find most helpful.

Background

More and more educational content is being delivered online, and many platforms rely on hosted discussion forums to ensure that students are engaging with their peers, faculty, and more generally, with the content. A 2007 article in the *Journal of Research on Education in Technology* argues that "asynchronous discussion tools can be used to integrate assessment activities that can help facilitate

[1] Noer, Michael. "One Man, One Computer, 10 Million Students: How Khan Academy Is Reinventing Education."
[2] "Formative vs Summative Assessment." *Enhancing Education - Carnegie Mellon University*. Carnegie Mellon University.
[3] Dringus, Laurie P., and Timothy Ellis. "Using Data Mining as a Strategy for Assessing Asynchronous Discussion Forums."

meaningful learning", but also that "there is a need to identify effective assessment methods appropriate to online learning."[4] Through interviews with online students, the researchers discovered that the students believed "most learning takes place" through asynchronous discussions.[5] However, totally unstructured discussions may not be the answer; a 2005 study found that "guidelines that assisted the facilitation and evaluation of online discussions increased the cognitive quality of student postings, promoting a deeper and more meaningful understanding of course content."[6]

The challenge then becomes how to build algorithms that actively and automatically provide structure and assist with the evaluation of online discussion. Although there has been promising work in the field of automated essay scoring[7, 8] (note that this is already in use by assessment provider ETS[9]), we limited the scope of our analysis to short-answer comments and discussions.

There have been a number of studies in literature applying machine learning to the goal of automating essay and comment grading. A 2014 paper out of Oxford found that applying Convolutional Neural Networks to text led to high performance on a number of common NLP tasks (sentiment analysis and question type prediction).[10, 11] In addition to CNNs, we use a number of other methods commonly applied to NLP tasks, such as feed-forward neural networks[12], Naive Bayes[13], and support vector machines[14].

Approach

Data

To collect the data for the project, we scraped about 1.1M comments from Khan Academy, an educational platform where individuals can watch videos on a variety of subjects ranging from math to literature to SAT Prep. These videos have comments enabled to allow individuals to ask questions, which others can answer. Individuals can choose to "vote" for replies that they like. We used the presence of votes as a rough proxy indicating a high-quality answer, although this assumption ignored the fact that quality assessments are subjective and vary widely between people (see the Conclusions section for more detail). We thought that this data would provide a starting point from which we could analyze comment quality, even given the caveat that comments on Khan Academy may be slightly different than comments in other contexts.

An example of a question and answer with votes on the Khan Academy video "Mars: Ancient Observations | Orbital Cycles"[15]:

[4] Vonderwell, Selma, Xin Liang, and Kay Alderman. "Asynchronous Discussions and Assessment in Online Learning."
[5] Vonderwell
[6] Gilbert, Patricia K., and Nada Dabbagh. "How to structure online discussions for meaningful discourse: a case study."
[7] Reilly, Erin D., Erin Stafford, Kyle Williams, and Stephanie Brooks Corliss. "Evaluating the Validity and Applicability of Automated Essay Scoring in Two Massive Open Online Courses."
[8] Song, Shihui, and Jason Zhao. "Automated Essay Scoring Using Machine Learning".
[9] "Automated Scoring and Natural Language Processing." *ETS Research.*
[10] Kalchbrenner, Nal, Edward Grefenstette, and Phil Blunsom. "A Convolutional Neural Network for Modelling Sentences."
[11] Zhang, Ye, Byron C. Wallace. "A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification."
[12] Goldberg, Yoav. "A Primer on Neural Network Models for Natural Language Processing."
[13] Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schutze. "Text Classification and Naive Bayes."
[14] Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schutze. "Support Vector Machines and Machine Learning on Documents."
[15] "Orbital Cycles". *Khan Academy.*

Baseline
For our baseline, each member of the team acted as a "human scorer" and manually labeled a set of 100 randomly selected comments as "high-quality" or not. This proved an indication of the complexity of the problem: each team member's labeling agreed with the source material between 54% and 61% of the time. Nor were we internally consistent: the three human scores were only in agreement with each other 56% of the time. (We would expect complete agreement 25% of the time through random chance alone.)

Code: https://github.com/kyleiwaniec/w266_Project/blob/master/hand-scores.ipynb

Modeling
We used a "grid-search" approach to modeling, exploring a number of different cuts of the dataset to find the most easily classifiable set of features.

We built models using support vector classifiers and Naïve Bayes. We also explored running our models on a large Multilayer Perceptron Neural Network (using scikit-learn) and a Convolutional Neural Network (using Theano and TensorFlow).

Code: https://github.com/kyleiwaniec/w266_Project/tree/master/models
Code: https://github.com/kyleiwaniec/w266_Project/tree/master/cnn-text-classification-tf

Features
We used a variety of features common in NLP: simple bigram counts, word counts, and length, as well as more complex features like Long Short-Term Memory likelihood[16]. The LSTM was a language-based model in which we tested whether a given comment was more similar to a typical high-quality or low-quality comment.

Code: https://github.com/kyleiwaniec/w266_Project/tree/master/models/features

Data Cuts
Finally, we tried to improve our results by varying the outcome and the set of data included in the model.

- For the dataset All Comments, we used a balanced dataset sampled from the 1.1M comments. The classes were based on the hasVotes variable, which allowed us to differentiate between comments that had received 1 or more votes from comments that had received no votes.
- One downside of the All Comments dataset is that it didn't allow for any differentiation within the positive class – it treated all comments that had received votes equivalently, whereas

[16] Hochreiter, Sepp, Jurgen Shmidhuber. "Long Short-Term Memory."

presumably comments with more votes are higher quality than comments with fewer votes. For the Relative Rank dataset, we tried to predict the relative vote rank of the comments, to determine whether a given comment A had more or fewer votes than comment B.

- The Top Ranked dataset was trained on the top 20% of comments for a particular video. This would ideally help differentiate the "best" comments from the "slightly better" comments. The downside of this approach was that many videos have very few (<5) comments.
- The Original + Reply dataset fed the model the combined string of the comment (usually a question) and the reply, in the hope that the question would provide valuable context about the answer. We were particularly optimistic about this approach using the CNNs.
  Code: https://github.com/kyleiwaniec/w266_Project/blob/master/ConvNet.ipynb
- The Reply only dataset included only comments that were "replies" to other comments, with the hypothesis that different features characterize a high-quality question than a high-quality answer.

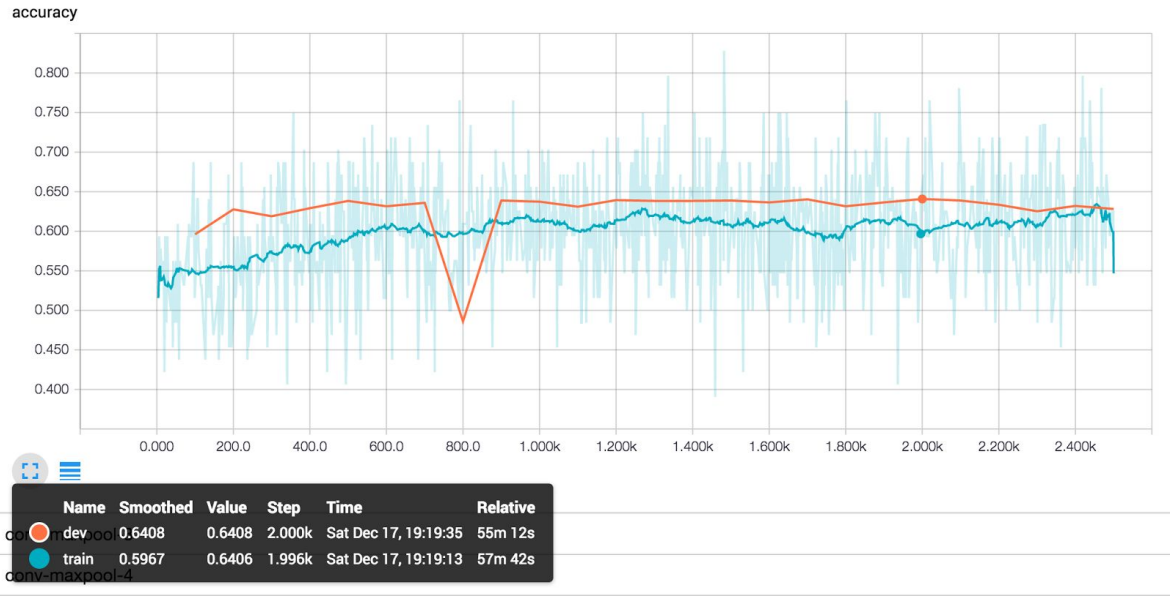
Results

The results below show the results for the best two models per dataset in terms of the F1-score.

Dataset	Features	Model	Precision	Recall	F1-score
Top ranked	Total comments	Convolutional Neural Network	88%	66%	0.76
Top ranked	Total comments	Multilayer Perceptron Neural Network (large)	61%	97%	0.75
All comments	LSTM likelihood	Support vector classifier	55%	93%	0.69
All comments	Length only	Support vector classifier	57%	83%	0.68
Relative rank	Length only	Support vector classifier	50%	100%	0.67
Relative rank	Bigram counts + length	Multilayer Perceptron Neural Network (small)	50%	100%	0.67
Replies only	Total comments	Support vector classifier	50%	100%	0.67
Replies only	Total comments + length	Support vector classifier	50%	100%	0.67
Reply & original	Word counts	Multilayer Perceptron Neural Network	56%	74%	0.64
Reply & original	Word counts	Multilayer Perceptron Neural Network (large)	56%	73%	0.63

We achieved our highest F1-scores using Convolutional Neural Networks on the Top Ranked data. See Appendix II for more details about the implementation of the CNN.

Code: https://github.com/kyleiwaniec/w266_Project/tree/master/CNN-results
Results: https://github.com/kyleiwaniec/w266_Project/blob/master/CNN-results/cnn-results.xlsx



Note that for many of these models the recall scores are very high, up to 100%. This relates to an earlier problem: many of the videos have only a small number of comments, so it is much more likely for a comment to end up in the "Top Ranked" portion even if it is not a high-quality comment. The high recall scores suggest that the algorithm is erring towards assuming comments are in the top.

While building the CNNs, we noted that modifying parameters such as embedding dimension, filter sizes, and number of filters had little effect on the results. We did not add any L2 regularization, because we never found the model to be overfitting. If anything, the accuracy on the dev set was usually better than on the train set. Additionally, results were best when no preprocessing (such as tokenization, lemmatization, etc.) was performed on the text itself beyond splitting on spaces. Though this seems counterintuitive, preserving all nuances in the text worked best.

Finally, we were also interested in understanding what features are helpful in predicting high-quality comments. Unsurprisingly, length alone does very well given how simple it is. This suggests that individuals tend to rely on relatively simple heuristics as indicators of comment quality when voting. Also, it is worthwhile to note that while the best models are complex neural networks, these do not outperform simple Naive Bayes models by as much as we thought they would (the Top Ranked dataset using total comments + length achieved an F1-score of 0.73 using Naive Bayes).

Conclusions

Our struggle to improve over the baseline despite applying complex methods is a reflection of the fundamentally challenging nature of this problem. The most significant problem is that "quality" is a highly subjective idea that varies widely between people. One student might prefer a detailed answer with a large number of external references or cited sources, whereas another student might prefer a concise answer that answers their specific question and nothing more. This subjectivity is perhaps best revealed by the lack of internal consistency among our team in our manual tagging exercise.

Secondly, the number of votes may have ultimately been a poor proxy for comment quality. This is consistent with research conducted out of Cornell University that framed large-scale voting mechanisms as fundamentally social activities: "User-provided helpfulness votes can highlight the most useful responses, but voting is a social process that can gain momentum based on the popularity of responses and the polarity of existing votes."[17] Users are subject to a number of different biases within the process of voting that affect their assessment of the quality of a comment. The Cornell study focused on the code-sharing platform StackOverflow, which has a very similar voting mechanism as Khan Academy. It is not at all unlikely that the same pattern would apply to the Khan Academy data.

[17] Lee, Moontae, Seok Hyun Jin, and David Mimno. "Beyond Exchangeability: The Chinese Voting Process."

Appendix I - Sources

References - Methods

- Goldberg, Yoav. "A Primer on Neural Network Models for Natural Language Processing." Draft. <http://u.cs.biu.ac.il/~yogo/nnlp.pdf>.
- Hochreiter, Sepp, Jurgen Shmidhuber. "A Convolutional Neural Network for Modelling Sentences." *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics.* Baltimore, Maryland, USA. Association for Computational Linguistics, n.d. Web. <http://www.aclweb.org/anthology/P14-1062>.
- Hochreiter, Sepp, Jurgen Shmidhuber. "Long Short-Term Memory." *Neural Computation.* 9(8) (1997):1735-80. Web. <http://deeplearning.cs.cmu.edu/pdfs/Hochreiter97_lstm.pdf>.
- Kalchbrenner, Nal, Edward Grefenstette, and Phil Blunsom. "A Convolutional Neural Network for Modelling Sentences." *Proceedings of the 29th Conference on Neural Information Processing Systems.* Barcelona, Spain. Web. <http://www.aclweb.org/anthology/P14-1062>.
- Kim, Yoon. "Convolutional Neural Networks for Sentence Classification." *New York University*. Web.
  Paper: <https://arxiv.org/pdf/1408.5882v2.pdf>.
  TensorFlow Implementation: <https://github.com/dennybritz/cnn-text-classification-tf>.
  TensorFlow Tutorial: <http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/>.
- Lei, Tao, Regina Barzilay, and Tommi Jaakkola. "Molding CNNs for text: non-linear, non-consecutive convolutions." *Computer Science and Artificial Intelligence Laboratory Massachusetts Institute of Technology*. Web.
  Paper: <https://arxiv.org/pdf/1508.04112v2.pdf>.
  Theano Implementation: <https://github.com/taolei87/text_convnet>.
- Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schutze. "Text Classification and Naive Bayes." *Introduction to Information Retrieval*: Cambridge UP, 2009. 253-87. Web. <http://nlp.stanford.edu/IR-book/html/htmledition/text-classification-and-naive-bayes-1.html>.
- Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schutze. "Support Vector Machines and Machine Learning on Documents." *Introduction to Information Retrieval*: Cambridge UP, 2009. 319-48. Web. <http://nlp.stanford.edu/IR-book/html/htmledition/text-classification-and-naive-bayes-1.html>.
- Zhang, Ye, Byron C. Wallace. "A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification." *Stanford*. Web. <https://arxiv.org/pdf/1510.03820v4.pdf>.

References - Background and Domain

- "Automated Scoring and Natural Language Processing." *ETS Research.* <https://www.ets.org/research/topics/as_nlp/>.
- Dringus, Laurie P., and Timothy Ellis. "Using Data Mining as a Strategy for Assessing Asynchronous Discussion Forums." *Computers & Education* August 45.1 (2005): 141-60. *Science Direct*, 20 July 2004. Web. <http://www.sciencedirect.com/science/article/pii/S0360131504000788>.
- "Formative vs Summative Assessment." *Enhancing Education - Carnegie Mellon University.* Carnegie Mellon University. <http://www.cmu.edu/teaching/assessment/howto/basics/formative-summative.html>.

- Gilbert, Patricia K., and Nada Dabbagh. "How to structure online discussions for meaningful discourse: a case study." *British Journal of Education Technology.* 36.1 (2005): 5-18. Web. <https://pdfs.semanticscholar.org/b89f/3d846b4f2d8ac91096efd93762d3cd773a0c.pdf>.
- Lee, Moontae, Seok Hyun Jin, and David Mimno. "Beyond Exchangeability: The Chinese Voting Process." *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics.* Baltimore, Maryland, USA. Association for Computational Linguistics, n.d. Web. <https://arxiv.org/pdf/1610.09428v1.pdf>.
- Noer, Michael. "One Man, One Computer, 10 Million Students: How Khan Academy Is Reinventing Education." *Forbes*. 19 Nov. 2012. Web. <http://www.forbes.com/sites/michaelnoer/2012/11/02/one-man-one-computer-10-million-students-how-khan-academy-is-reinventing-education/#452015633c05>.
- Reilly, Erin D., Erin Stafford, Kyle Williams, and Stephanie Brooks Corliss. "Evaluating the Validity and Applicability of Automated Essay Scoring in Two Massive Open Online Courses." *The International Review of Research in Open and Distributed Learning.* 15.5 (2014). Web. <http://www.irrodl.org/index.php/irrodl/article/view/1857/3067>.
- Song, Shihui, and Jason Zhao. "Automated Essay Scoring Using Machine Learning". *Stanford.* <http://nlp.stanford.edu/courses/cs224n/2013/reports/song.pdf>.
- Vonderwell, Selma, Xin Liang, and Kay Alderman. "Asynchronous Discussions and Assessment in Online Learning." *Journal of Research on Technology in Education* 39.3 (2007): 309-28. Web. <http://files.eric.ed.gov/fulltext/EJ768879.pdf>.

Appendix II - Convolutional Neural Network Implementation Details

Theano implementation

GPU 8 cores, 15GB RAM
https://trial.dominodatalab.com/u/kylehamilton/rcnn266/settings#execution

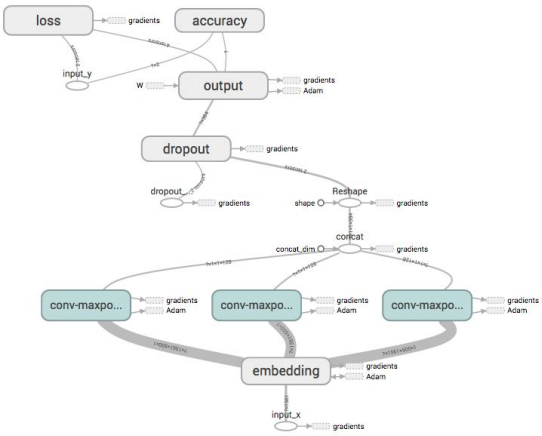
run#9 14h, 43m, 27s Dec 10, 2016 @ 12:16 am

Namespace(act='relu', batch=16, decay=0.5, depth=2, dev='data/theano_data_dev', dropout_rate=0.3, embedding='word_vectors/stsa.glove.840B.d300.txt.gz', eval_period=-1, hidden_dim=300, l2_reg=1e-05, layer='strcnn', learning='adam', learning_rate=0.001, load='', max_epochs=100, mode=1, order=2, pooling=1, save='output_model_ka', test='data/theano_data_test', train='data/theano_data_train')
18240 pre-trained embeddings loaded.
layer 0: n_in=300 n_out=300
layer 1: n_in=300 n_out=300
layer 2: n_in=600 n_out=2
total # parameters: 541800

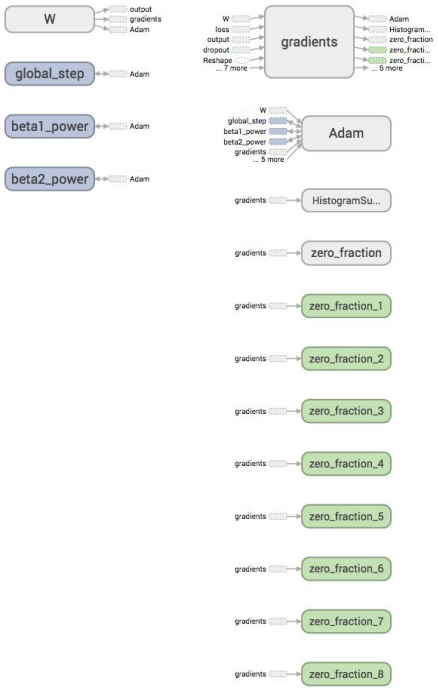
Best Result:
Epoch 2.0 loss=0.6598 |g|=0.396705782415 [249.85m]
['0.58', '8.75', '13.54', '9.38', '0.63', '2.54', '2.85', '1.98', '1.88']
dev accuracy=0.6877 best=0.6877
test accuracy=0.6098

TensorFlow Implementation Details

Main Graph



Auxiliary nodes



DROPOUT_KEEP_PROB=0.5
EMBEDDING_DIM=128
FILTER_SIZES=3,4,5
L2_REG_LAMBDA=0.0
NUM_EPOCHS=200
NUM_FILTERS=128

positive_examples: 248,241
negative_examples: 163,887

Max_document_length: 702

Vocabulary Size: 173,722
Train/Dev split: 410,068/2,060

TopRank dataset
precision: 0.881595882
Recall: 0.661356505
F1: 0.755757827
Accuracy: 0.6458

Replies only dataset
Precision: 0.685344828
Recall: 0.60861244
F1: 0.644703497
Accuracy: 0.541830065