Project Title
Akshay Agrawal, Shane Leonard
On Knowledge and Confusion: Classifying Text Documents in MOOC Discussion Forums

Introduction:
Interest in educational data mining (EDM) has grown exponentially in recent years [1]. The Journal of Educational Data Mining was established to accommodate the surge in EDM research activity; in the following years, Learning at Scale and the International Conference on Learning Analytics cropped up as well. Broadly, EDM "is concerned with methods to explore unique types of data in educational settings and … to better understand students and the settings in which they learn," [1]. Its applications cut across many domains within the educational setting, ranging from visualization of course activity to automating course construction.

The advent of Massive Open Online Course (MOOC) platforms edX and Coursera in 2012 unleashed vast troves of data for researchers to mine. Much effort has gone into developing "student models" — predicting students' next- question-accuracy and modeling their acquired skills. Some argue that such research is reaching the point of diminishing returns [2].

In contrast, relatively little research has been devoted to digging into MOOC discussion forums. We find ourselves interested in surfacing relevant content for instructional staff — when it comes to MOOCs, it's intractable for staff to sift through a forum's sea of threads, notes, and comments.
We make the (we think reasonable) assumption that instructional staff are particularly interested in addressing deficiencies in students' understanding. As such, we attempt to surface discussion forum posts that exhibit *confusion.* We say that a post exhibits confusion if it appears that its author either explicitly requests for clarification about a course topic, or if his language implicitly reveals a gap in comprehension. In contrast, we say that a post exhibits *knowledge* if it delivers factual information relevant to the topics studied in the course. If the post conveys neither confusion nor knowledge, we say that it is *neutral*. While EDM work on analyzing sentiment in MOOC forums exists [3], to the best of our knowledge, no previous work has explicitly attempted to identify confusion in posts.

We frame the problem of retrieving confused posts as a multilabel classification problem, of the following form: Given the body of a discussion forum post *P* with a true unknown label *L* in *{ knowledgeable, neutral, confused}*, apply some hypothesis *h* that correctly divines *L*. That the problem is a multilabel one is of significance: solving the easier binary classification problem in which posts are labeled as *confused* or *not-confused* would entail forfeiting potentially useful information about where a post's particular author lies on the knowledge-confusion spectrum.

The Nature of the Data
In this paper, we focus exclusively on discussion forum belonging to courses offered through Stanford's Open edX MOOC initiative.

Three sets of nearly 10,000 distinct discussion forum posts each — from courses Statistics in Medicine, Statistics 216, and How to Learn Math — were collected. Every set was judged by three distinct judges, for a total of nine judges, and each post was scored along six dimensions, one of which we currently make use of. We describe data collection and gold set creation in detail in a separate document.

Each set thus has approximately 10,000 rows, each row a distinct forum post. There are 2 columns: (1) To what extend does the post communicate knowledge or confusion (encoded as a number from 1 through 7, with 1 indicating maximum plentiful knowledge and 7 indicating plentiful confusion) and (2) What is this post's forum_post_id in the EdxForum database that lives on datastage.stanford.edu? Column (2) allows us to retrieve metadata about the post's author and parent thread; although we do not yet make use of this information, we plan to do so in the future.

Feature Representation: Modeling a Discussion Forum Post

We have a potentially |**D**|+k-dimensional feature space, where |**D**| is the number of unique words encountered in a training set (on the order of 10s of thousands) and k is the number of additional features we engineer. Both |**D**| and k are functions of the particular feature selection pipeline that we select in a given classification run.

Our feature selection pipeline consists of three phases: (1) text pre-processing, (2) feature engineering, and (3) feature extraction. Each phase consists of a number of policies that we are currently experimenting with; in particular, we can chain together any combination of policies in (1), (2), and (3) to construct a unique classification pipeline.

In phase (1), we implement a policy that maps all numbers to a single token, and a policy that allows for crafting a custom, domain-specific stop- word-list. We also intend to implement policies that map contractions (expanded and unexpanded) to unique tokens, and that maps all mathematical equations to a single unique token.

In phase (2), we implement a policy that inflates the counts of words in the first sentence (with the assumption that students summarize their intent early on in their paragraph(s)), and a policy that inflates the counts of all words occurring in noun phrases. We intend to make both of these heuristics more robust, and to engineer domain-specific features (e.g., place extra emphasis on words belonging to a "MOOC confusion" lexicon, and add metadata about the post's author).

Finally, in phase (3), we map term frequencies to integer counts and provide policies to tokenize terms in a custom fashion, rescale counts using tf-idf and select the *n* most informative features, as determined by a chi-squared test.

We are currently still experimenting to determine the best combination of feature selection policies. If the BOW approach does not prove promising, then we might choose an alternate kernel-based representation that maps documents to all subsequences of a particular length [4].

Classification Models

We have discrete inputs — term counts — and a categorical output — knowledgeable, neutral, or confused. We're currently experimenting with multinomial naive Bayes, multinomial logistic regression, and linear support vector machines, since these three models 1) are often cited in text classification literature and 2) require relatively little effort to get off the ground.

We're building our models on top of scikit learn.

Our original approach was a simple bag of words, with no document preprocessing. Our current leading approach implements several different strategies. The document is first preformatted by replacing all numbers with a placeholder, and then it is tokenized, treating punctuation and single-letter words as valid tokens. We use a modified English stop words list that doesn't treat 'I' as a stop word. We reduce the feature space by applying chi-squared Univariate Feature Selection. For the SVM we apply L1 regularization with a penalty parameter of 0.5 to reduce overfitting. Note that the generalization error improved for every label, even though the feature space has been significantly reduced.

Model	Label	Train: F1	Test: F1
SVM	knowledgeable	0.986599	0.389445
SVM	neutral	0.996483	0.864509
SVM	confused	0.993741	0.35207
Naive Bayes	knowledgeable	0.737699	0.441833
Naive Bayes	neutral	0.928654	0.888782
Naive Bayes	confused	0.711252	0.367184
Logistic Reg	knowledgeable	0.91259	0.420034
Logistic Reg	neutral	0.974787	0.882031
Logistic Reg	confused	0.911703	0.378374

Table 1: Original Bag-of-Words Approach

Model	Label	Train: F1	Test: F1
SVM	knowledgeable	0.672791	0.396975
SVM	neutral	0.922132	0.867997
SVM	confused	0.701643	0.35299
Naive Bayes	knowledgeable	0.653983	0.487236
Naive Bayes	neutral	0.908834	0.879653
Naive Bayes	confused	0.648954	0.466741
Logistic Reg	knowledgeable	0.664368	0.421321
Logistic Reg	neutral	0.918744	0.873467
Logistic Reg	confused	0.675589	0.382527

Table 2: Full Approach