Evaluation of essays using incremental training for Maximizing Human-Machine agreement

Dhananjay Ambekar
123059008

Guided By
Prof. D. B. Phatak

Department of Computer Science and Engineering
Indian Institute of Technology, Bombay

June, 2014
Introduction to topic

Areas of Interest

- Massive open on-line courses (MOOCs)
- Grading essays in MOOCs
- Automated essay grading
- Correlation between Human-Machine grading
Self & Peer Assessments
An Introduction

- **Self Assessments**
  Students grade assignments or tests based on a teacher's guidelines. *Rubrics* are used.

- **Peer Assessments**
  Student grades submissions of other students after his completion. A process for calibration may or may not be applied.
AI Grading
aka ML Grading

- MOOCs have large number of submissions with varied data, ML grading an apt choice
- Uses Machine Learning techniques along with NLP for text prepossessing
- Part of the data is graded by instructor and same is used to train models for grading further

**Figure:** Process of automatic Evaluation of essays
Discussion on automated Scoring

- Essays are subjective in nature, best sources to judge understanding of student
- Multiple graders will produce different grading for same data
- Produces inconsistency among grades, Automated models are consistent (might not be accurate)
- If trained well, automated scoring is consistent with human graders
<table>
<thead>
<tr>
<th>Essay from location X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cow is a Domestic animal found all over the world. They are raised for milk and Dairy products. They are major source of consumable Beef &amp; leather.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Essay from location Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cow is one of the most useful Domestic cattle in the world. They give us milk and Dairy.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Essay from location Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>Every Farmer has cows. They use cows to milk and Dairy products. In India Cow is worshipped. It has mentions in Holy texts.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Essay from location Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cattle are venerated within the Hindu religion of India. They provide us milk and Dairy.</td>
</tr>
</tbody>
</table>
Objective is to achieve the accuracy as near as human Evaluation.

- **Features contained in responses** differ adding diversity to them. (For an example - For Indian students, In an Essay written on Cow, *People worship Cow.* is an obvious feature.)

- These Features might not be captured while training by picking samples randomly because with respect to large number of students, frequency of these features might be very less and **Evaluation Rubrics** may or may not capture that content detail.

- While in training phase, another Iteration of training itself can be added to either **learn** or to **add** such features
Auto Evaluation

Figure: Flow of Incremental Experimentation
Estimating accuracy against Human grading

Cohen Kappa Coefficient

\[ k = \frac{P(a) - P(e)}{1 - P(e)} \]

Pr(a) is the relative observed agreement among raters
Pr(e) is the hypothetical probability of chance agreement

**Precision**

\[ P = \frac{\text{No. of cases of agreement}}{\text{No. of evaluations by Human} + \text{No. of auto evaluations}} \]
Approach & Implementation

- Automated evaluation system
- Formulate a process for Iterative training & evaluation
- Comparing performances using standard measures
- Django Application compatible with EdX
- Python-Scikit & language toolkit for computational Purpose
Django framework is MVT (Model-View-Template) based python framework used to develop complex and database driven web applications. Structure of a typical Django web application

- **Model** - It is a python module that is used to design represent database entities as well as synchronize them throughout the application.

- **Views** - A Django application can have multiple views containing several methods. These methods in turn serve user requests (HttpRequests) for content or any other functions operating on data.

- **Templates** - They serve as a user interface for Models-Views designed in an application.
Data Needed For Application

This data may not be directly used from database but through APIs or XML object files which are used in (Existing) process of evaluation. For this project purpose, Data is accessed through django models created in applications.

Data is attributed in to two main components

1. Student Data which contains all the student profile information which are registered to the course.
   Table: AUTH_USER
   Table: AUTH_USER_PROFILE
   Table: STUDENT_COURSEENROLMENT
   Table: USER_ID_MAP

2. Course-ware Data contains course content as well as assessment/submission details.
   Table: COURSEWARE_DATA
1. Fetch the required data - Submissions consisting of textual data are fetched from Database according to the requirement. This module also divides the submissions according to the student properties. (For Example in this case we have divided the data according to student geographical locations.)

**Figure:** Retrieving required Data
2. Operate on data for Training and Evaluation
This Evaluation part was designed to work in pipe-lined manner as in involved multiple stages such as Feature Extraction, Training, Testing and Performance measurement.

- Text Vectorizer Convert a collection of text documents to a matrix of token counts. Fit_transform function Learn the vocabulary dictionary and return term-document matrix which later used to train classifier.

- The Python view then retrieves parameters required for the Classifier specified (class sklearn.naive_bayes.MultinomialNB) and start operating on data-sets generated above.

- After automated evaluation is performed the results and performance measurements are written in the designated directories.
### Results

**Table: Evaluation Details by all Techniques**

<table>
<thead>
<tr>
<th>Grade</th>
<th>Application</th>
<th>EDX-AI</th>
<th>Human</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>5</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>22</td>
<td>27</td>
</tr>
<tr>
<td>4</td>
<td>41</td>
<td>35</td>
<td>34</td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td>18</td>
<td>21</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>13</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>100</strong></td>
<td><strong>100</strong></td>
<td><strong>100</strong></td>
</tr>
</tbody>
</table>
Total 10 samples were examined to assign either *Good* or *Poor* label.

<table>
<thead>
<tr>
<th>Total 10 samples</th>
<th>Grader II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good</td>
<td>6</td>
</tr>
<tr>
<td>Poor</td>
<td>2</td>
</tr>
<tr>
<td>Grader I assessment</td>
<td>Good</td>
</tr>
<tr>
<td></td>
<td>Poor</td>
</tr>
</tbody>
</table>

*Table: Inter grader comparison*

**Cohen’s kappa Coefficient**

\[
P(a:\text{agreement}): \frac{(6+2)}{10}
\]

\[
\Pr(\text{e:random agreement}): = 0.48 + 0.08 = 0.56.
\]

\[
k= \frac{(0.8-0.56)}{(1-0.56)} = 0.56
\]
Grading Matrix

**Table: Example Grading Distribution Human-Edx AI**

<table>
<thead>
<tr>
<th>Grades</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>18</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>28</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>25</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>25</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>13</td>
<td>2</td>
<td>0</td>
<td>25</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>8</td>
<td>18</td>
<td>43</td>
<td>13</td>
<td>14</td>
<td>4</td>
<td><strong>100</strong></td>
</tr>
</tbody>
</table>
### Agreement between Human Grader and Edx-AI Evaluation

- **Observed proportionate agreement**: $0.75 \pm 0.04$
- **Cohen’s Kappa Correlation**: $0.682$

### Agreement between Human Grader and App-Evaluation

- **Observed proportionate agreement**: $0.78 \pm 0.04$
- **Cohen’s Kappa Correlation**: $0.717$
observations
Conclusion

- Geographical characteristics affect textual responses
- Automated Evaluation is a fair Choice in MOOCs
- If taken account we can perform Improved grading
  Transparent Feedback
Future Work

- Integration with Open edX along with Existing AI-Evaluation work-flow as a Django application
- Distinguish cases of students where Human Grader-Automated Evaluation agreement is very less
- Can be used to understand how students belonging to particular category (Ex. geography), culture have learned & gained knowledge of the subject compared to other peers.
Thank You.


An evaluation of phrasal and clustered representations on a text categorization task.  

Building a large annotated corpus of english: The penn treebank.  


[13] NLTK.  
Natural language toolkit, 2014.  
Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. OReilly Media Inc.

[14] POS.  
Pos tagging, 2014.  
[http://www.comp.leeds.ac.uk/ccalas/tagsets/upenn.html].

Handling re-grading of automatically graded assignments in moocs.  

Taylor & Francis, 2002.

An overview of current research on automated essay grading.  

Additional References with citations to content are listed in corresponding Report
Evaluation Rubrics

- Rubric means “a standard of performance for a defined population”
- Rubrics are the way of communicating expectations from a problem/assignment which student will attempt
- Defined by Instructor at the time of creating a problem, and are consist of different criterion which will try to understand the submission made by a student
Steps in Feature Extractions

- **Prepossessing of input data**
  Involves removal of non-ASCII characters. Processing punctuation such as period comma with spaces.
  In this step essay is truncated with some limit of words, for example essays with more than 2500 words will be truncated to this limit.

- **n-Grams**
  A Parts of Speech (POS) tree is created before deriving n-grams.

**Sentence:**
“Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.”

**Tokens:**
[Pierre / Vinken / , / 61 / years / old/ , / will / join / the/ ...continued]
Steps in Feature Extractions

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NNP</td>
<td>noun, proper, singular</td>
</tr>
<tr>
<td>VB</td>
<td>verb, base form</td>
</tr>
<tr>
<td>JJ</td>
<td>adjective or numeral</td>
</tr>
</tbody>
</table>

POS-tagged tree example
Steps in Feature Extractions

- **Synthetic training examples** Many essay evaluation processes also take account of synonyms of tokens generated (Ex. NLTK)
- **Feature Vector** Features are maintained for each essay in data set
  - Spelling Error Features: Avg. spelling errors per essay
  - Grammar Error Features: Counts of good n-Grams
  - Bag of stemmed words
Bayesian Independence Classifiers

These probabilistic classifier estimate a probability of document being in particular category provided that certain features are present in the document.

\[
\log P(C|D) = \log P(C) + \sum_i \begin{cases} 
\log \left( \frac{P(A_i|C)/P(A_i)}{P(\bar{A}_i|C)/P(\bar{A}_i)} \right) & \text{if } D \text{ contains feature } A_i \\
\log \left( \frac{P(\bar{A}_i|C)/P(\bar{A}_i)}{P(A_i|C)/P(A_i)} \right) & \text{if } D \text{ does not contain feature } A_i 
\end{cases}
\]

\(P(C)\) is probability that a document belongs to class \(C\), a prior probability. Class \(C\) can be a class/category of Good,Average or any class which is defined earlier while process of training. 
\(P(A_i|C)\) is the conditional probability the document has feature \(A_i\) and given that it belongs to class \(C\). 
\(P(A_i)\) is a prior probability of any document containing feature \(A_i\). 
\(P(\bar{A}_i|C)\) is the conditional probability that document does not feature \(A_i\) given that class is \(C\). 
\(P(\bar{A}_i)\) is the probability that document does not have feature \(A_i\).
K-nearest Neighbour

- K- essays most similar to the test essay are selected
- Score is given to essay based on weighted average of k selected essays in test set
- Similarity is calculated by querying entire document against test set
Gradient Boosting

- Produces a prediction model in the form of an group of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

- **learn_rate**: (Value = 0.5)
  Learning rate shrinks the contribution of each tree by learning_rate. There is a trade-off between learning_rate and n_estimators.

- **max_depth**: (Value = 4)
  Maximum depth of the individual regression estimators. The maximum depth limits the number of nodes in the tree. Tune this parameter for best performance.

- **min_samples_leaf**: (Value = 3)
  The minimum number of samples required to be at a leaf node. Value = 3

- **max_features**: (Value = auto)
  The number of features to consider when looking for the best split.
## Summary of Performances

<table>
<thead>
<tr>
<th>System</th>
<th>Technique</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>PEG</td>
<td>Statistical</td>
<td>0.87</td>
</tr>
<tr>
<td>IEA</td>
<td>Algebra</td>
<td>85-91</td>
</tr>
<tr>
<td>ETS-I</td>
<td>NLP</td>
<td>93-96</td>
</tr>
<tr>
<td>BETSY</td>
<td>Bernoulli/NLP</td>
<td>80</td>
</tr>
<tr>
<td>Automark</td>
<td>NLP/ML</td>
<td>93-96</td>
</tr>
<tr>
<td>C-Rater</td>
<td>NLP/ML</td>
<td>80</td>
</tr>
</tbody>
</table>