SRI International

# LDC – Automated Essay Scoring using SRI's ATSE software

## Project Interim Report #3

**April 21, 2017**

**John Niekrasz**
Advanced Analytics Lab
Suite #203
9988 Hibert St.
San Diego, CA 92131

# Table of Contents

# 1 Introduction

This interim report describes progress to-date on the development of an LDC-customized version of SRI's ATSE software. The principal goal of the customization project is to improve the ATSE system's accuracy with respect to prompt-blind scoring of student responses to LDC writing prompts. This report marks the third main milestone of a planned year-long project, associated with the completion of Subtask 1.4 "SCALE Rubric Features." The project plan schedules this milestone for the ninth month of the project. The project is about one month behind schedule.

The goal of Subtask 1.4 is to improve ATSE by implementing system optimizations relevant specifically to LDC's use of the *SCALE rubric*. We anticipated that some feature extractors not yet implemented in ATSE would be identified as good candidates for obtaining this improvement. We estimated that we could garner a 5-10% relative improvement in scoring accuracy through execution of this task.

The following sections document our most recent work: "Progress summary," "February 2017 Scoring Dataset Analysis," and" SCALE Rubric." The remainder of the report includes content from previous interim reports that has been copied here for ease of reference. Also, an attached spreadsheet supplement contains further details on quantitative evaluations.

# 2 Progress summary

One of our most significant accomplishments in this period was the successful **integration of the recently collected hand-scored dataset** (which we refer to as the "February 2017" dataset) into our analysis pipeline. After having formatted and validated the data, and after writing the software necessary to convert the data into ATSE format, we are now exclusively working with a much larger combined dataset for ATSE model training purposes. This has expanded our training set from 7 prompts to 18, and from about 850 responses to 2600. Initial results show that the addition of the February 2017 data **improved the system scoring accuracy (on average by trait) by 3.0%,** taking the average accuracy by trait from 70.6% to 72.6% (relative to human scoring, and using the same version of ATSE that was used to generate results for the previous report). Unfortunately, this increase is not as significant as expected, and all signs point to poor inter-rater reliability as the main culprit in this less-than-expected improvement. (The average inter-rater reliability for the October 2016 data was .628, while it was .487 for the February 2017 data.) Nonetheless, the fact that no work has yet been done to remove unreliable training data

suggests there is room for leveraging this dataset to improve results even further. We have also yet to complete this experiment using our most recent version of ATSE.

The other main accomplishment achieved in this period was the successful **implementation of new SCALE rubric features.** Our experiments have shown that inclusion of these new features **improved the system scoring accuracy by between 7.9 and 10.2%, depending on the trait being scored.** In particular, we focused our efforts on exploring the relationship between grade level and rubric specification. We executed a differential scoring experiment to examine whether an across-grade-band model would perform better than a within-grade-band model and found that including grade level as an additional feature in an across-band model was the most powerful way of including that information into the system. The improvement had a significant impact, and as a result, **ATSE is now achieving, on average across traits, an accuracy level of 79.3% of human inter-rater scoring accuracy**.

## Discussion of next steps

Our main goal for this project is to achieve 90% human-relative scoring accuracy for prompt-blind scoring. With about two months remaining in the project, we propose that in order to achieve this goal, further effort ought to focus on feature additions and refinements of the scoring model, rather than on the currently planned Subtask 1.6 "Construct relevance." While this latter task is a necessary stepping stone toward producing inline feedback, which is a long term LDC goal, it is likely that this effort may actually reduce ATSE performance, since it will be focused on pruning out system features that are shown to be construct irrelevant.

We recommend exploring *data selection* as an alternative method of system feature improvement that could yield a more significant impact on achieving project objectives. That is, given the persistent difficulty of obtaining reliable human scoring (which is likely to persist beyond the life of this project), we recommend focusing effort on implementing automatic processes that can selectively remove (or weight) human scoring information, and perhaps even use individual scores to the system's advantage (as opposed to using consensus or average scores). In this way, the system would be able to better use available information coming from human scored datasets. SRI is interested in hearing LDC's feedback concerning these recommendations.

# 3 February 2017 Scoring Dataset Analysis

We conducted a study of the inter-rater reliability of the Feb 2017 dataset and compared results with those from the Summer 2016 data. The results are provided in the Supplement A spreadsheet under the "INTER-RATER BY PROMPT & TRAIT" and "INTER-RATER BY RATER" sheets. The first sheet shows inter-rater reliability scores by prompt and trait. The second sheet shows reliability by individual scorer. A table showing average agreement across traits for each prompt is reproduced here as Table 1.

Table 1. Inter-rater agreement by prompt.

| PROMPT | QKAPPA |
|---|---|
| ldcfeb2017_ms_social_studies_task_2_causes_of_american_revolution | 0.289 |
| ldcfeb2017_hs_social_studies_task_1_andrew_carnegie | 0.375 |
| ldcfeb2017_hs_science_task_1_nuclear_power | 0.410 |
| ldcoct2016_period_is_pissed | 0.439 |
| ldcfeb2017_hs_ela_task_2_ex_machina | 0.464 |
| ldcfeb2017_hs_social_studies_task_2_were_immigrants_welcome_ | 0.493 |
| ldcfeb2017_ms_science_task_1_fracking | 0.502 |
| ldcoct2016_nuclear_power | 0.503 |
| ldcfeb2017_ms_ela_task_2_dream_within_a_dream | 0.505 |
| ldcfeb2017_hs_science_task_2_bpa | 0.515 |
| ldcfeb2017_ms_social_studies_task_1_jamestown | 0.527 |
| ldcoct2016_water_in_colorado | 0.594 |
| ldcfeb2017_hs_ela_task_1_words_matter | 0.627 |
| ldcfeb2017_ms_ela_task_1_growing_up_is_hard_to_do | 0.657 |
| ldcoct2016_un_education | 0.662 |
| ldcoct2016_no_guitar_blues | 0.681 |
| ldcoct2016_cold_war_red_scare | 0.751 |
| ldcoct2016_fracking | 0.765 |
| Average | 0.542 |

Overall, inter-rater reliability with the new dataset was lower than with the previous dataset. Some prompts had particularly low values (notably MS SS Task 2 and HS SS Task 1, both of which were on average below 0.4 quadratic weighted kappa). Citation of evidence was the most reliable trait in both datasets, with Conventions being the least reliable in both datasets. Additionally, 8 out of 47 individual scorers (17%) had average agreement scores below 0.4.

# 4 SCALE Rubric

We also conducted a study exploring the role of grade level and differential application of the SCALE rubric. We had noted that the rubric is differentially applied across grade bands. Even more importantly, we noted that anchor papers across grade bands were markedly different. We therefore hypothesized that the rubric was being applied in rather different ways across the grade bands. However, as was noted by Nicole Renner, the intention is for the rubric to function on a continuum across grades, with differences in application mainly designed to capture growing complexity in students' application of the writing skill. We therefore hypothesized that scores across grade bands are most likely best treated within a single model, and that grade level should be incorporated as a feature.

We implemented this idea by adding a set of ATSE input features that can be derived from LDC prompt meta-information. Fortunately, adding these features allowed us to obtain significant improvements in system accuracy. On average across traits, the system improved 9.1% in comparison to not using the features. Table 2 below summarizes the results. More details, including results by trait, are included in the Supplement A spreadsheet under the "FEB 2017 RESULTS" sheet.

Table 2. ATSE system improvements due to Subtask 1.4 (SCALE rubric) features and addition of new data.

| | |
|---|---|
| JAN 2017 SYSTEM (ALL DATA) | 0.394 |
| JAN 2017 SYSTEM (2016 OCT DATA ONLY) | 0.427 |
| JAN 2017 SYSTEM RELATIVE (ALL DATA) | 0.726 |
| JAN 2017 SYSTEM RELATIVE (2016 OCT DATA ONLY) | 0.706 |
| 2017 FEB DATA ADDITION RELATIVE IMPROVEMENT | **3.0%** |
| APR 2017 SYSTEM (ALL DATA) | 0.429 |
| APR 2017 SYSTEM RELATIVE (ALL DATA) | **0.793** |
| SUBTASK 1.4 ADDITION RELATIVE IMPROVEMENT | **9.1%** |

With much of our time spent working to incorporate the new scoring data and analyzing inter-rater reliability issues, there are still some SCALE-specific features that could be implemented if time allows in the remaining portion of the project. For example, it is clear that the system, as well as humans, do not have a solid understanding of the "Conventions" trait. With some additional discussion of what this means in the context of various subject areas, and a

subsequent implementation of some corresponding features, ATSE might even garner an increase of more than 9%.

# 5 Summer Scoring Dataset Analysis

This section documents the Summer Scoring dataset and the conclusions we have drawn from using it in our first baseline evaluation. It is an updated and extended version of a preliminary report prepared in August 2016 with the help of Fannie Tseng of Preva Group.

## 5.1 Dataset collection and manual scoring

The Summer Scoring dataset is comprised of a convenience sample of student responses to LDC prompts in middle and high school ELA, Science and Social Studies, collected from numerous schools in the state of Colorado. The responses were manually scored by trained scorers, who were teachers recruited from Denver metropolitan area schools. The teachers were trained on the LDC rubric by expert facilitators from SCALE and public school districts in the Denver area. Scorers were trained using a hands-on calibration process designed by SCALE, in which:

- Teachers individually score sample papers;
- Teachers assemble in subject-specific groups for discussion about the scores, moderated by the subject-specific facilitator;
- The group tries to reach agreement in their scoring among the group and to scores recorded by the subject-area facilitator (called master scores).

This process is repeated multiple times, until individual scores are calibrated (i.e. in agreement). In addition, once the scoring session begins, the facilitators use "check papers" – papers that are scored by all the scorers in a subject area group, to continually monitor scores and make sure that they are calibrated.

## 5.2 Manual scoring descriptive statistics

The dataset collected for this study consisted of responses to seven (7) different prompts, with different sets of students responding to each prompt. The responses were then manually scored, with an average of 119 scored responses per prompt. Table 3 provides summary information about the number of scored responses for each of six grade-level – subject area pairings.

| Grade Level-Subject | # of Prompts | # of Responses Scored per Prompt | # of Responses with Multiple Scores | % of Responses with Multiple Scores |
|---|---|---|---|---|
| HS-ELA | 1 | 152 | 36 | 24% |
| HS-Science | 1 | 136 | 41 | 30% |
| HS-Social Science | 1 | 60 | 60 | 100% |
| MS-ELA | 1 | 105 | 10 | 10% |
| MS-Science | 2 | 132 | 32 | 24% |
| MS-Social Science | 1 | 116 | 69 | 59% |

Table 3. Summary statistics for the Summer Scoring dataset

## 5.3 Dataset preparation and pre-processing

Student response data was provided to SRI using Microsoft Word format. We note that the documents often have minor issues associated with formatting, as in Figure 1, which is a screen shot of an example submission in Word format. Note the unusual elision of some words when occurring after the word "I". Some of these formatting issues may originate with the original providers of the documents. Others we determined were due to the hand-copying of word documents during LDC's data preparation and anonymization process.

It is recommended that in future rounds of data collection, any manual transformation of data be avoided if possible. Instead, collecting input directly from students via a consistent digital interface is recommended. (See discussion of the Tao assessment platform in Section 10 below.) Note that ATSE requires plaintext input, and we apply text extraction scripts to convert the Microsoft Word documents to ASCII plaintext.

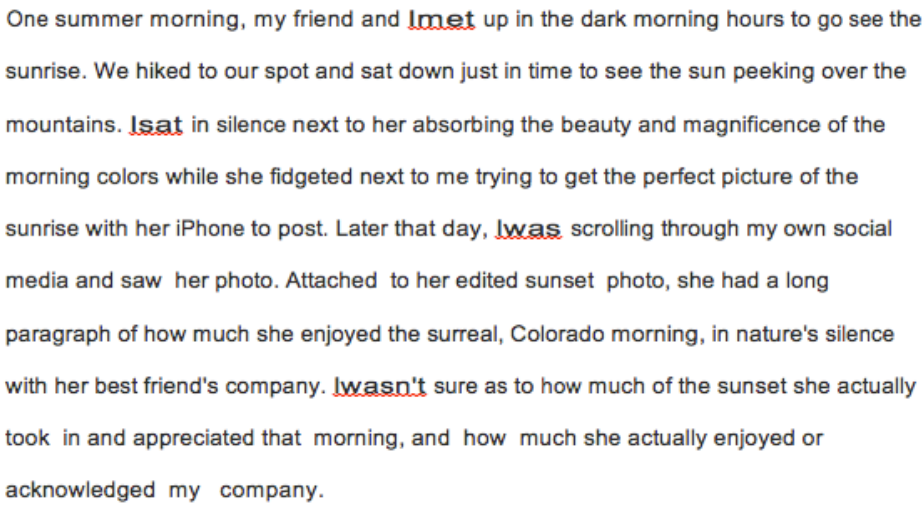


Figure 1. An example of a formatting error in a paragraph from one of the Microsoft Word documents

Scoring information was provided to SRI in the form of tab-separated-value (TSV) files. There were no issues in pre-processing these files as they conformed to the required formats documented in the ATSE manual, which was provided to LDC data in advance.

## 5.4 Inter-rater reliability

The inter-rater reliability of the Summer Scoring dataset is shown in Figure 2. Each data point characterizes the reliability for a particular prompt-trait combination. Approximately half of the scoring sets would be considered at least a "good" level of agreement, while the other half are characterized as "moderate" or perhaps "poor" in some cases.

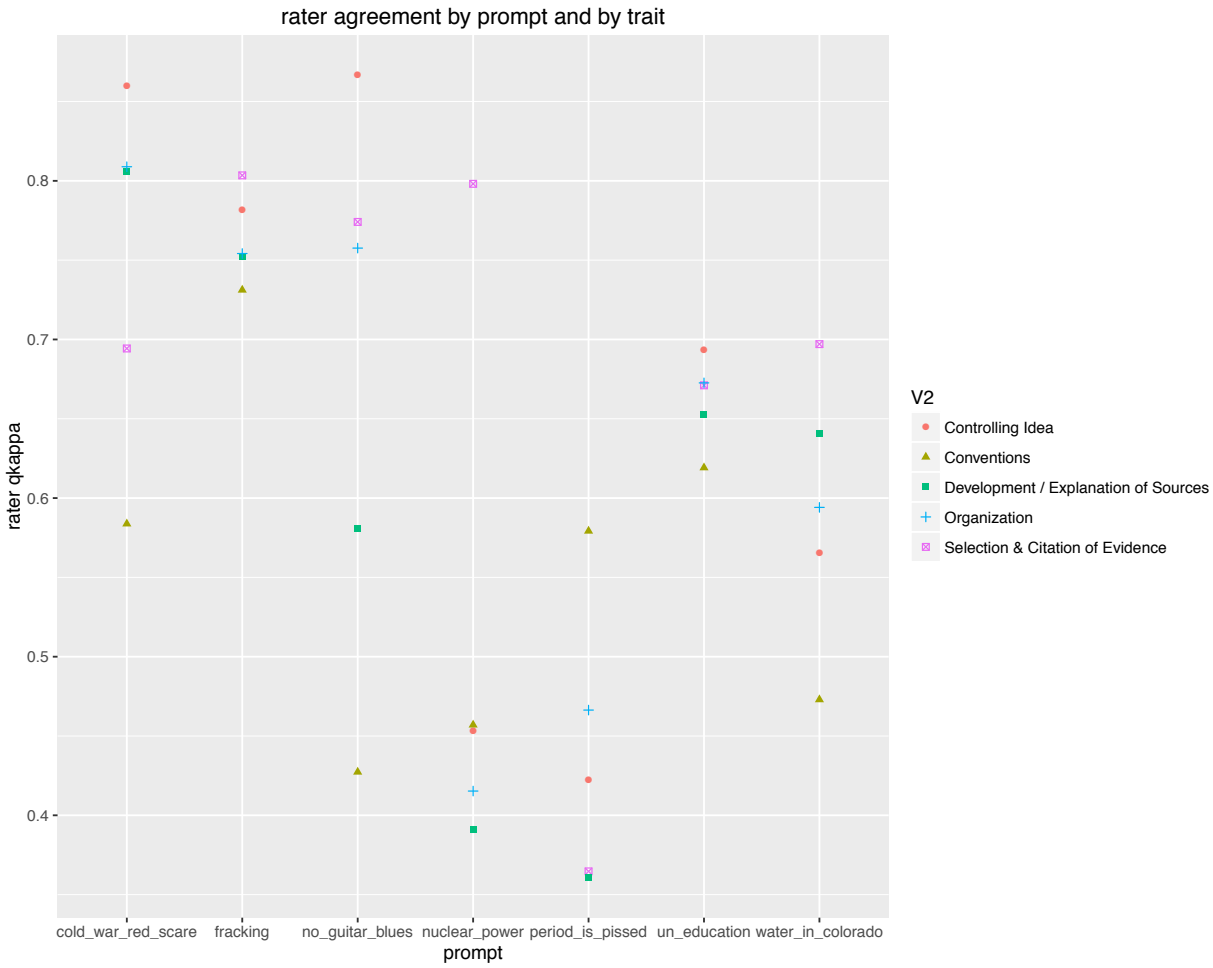


Figure 2. Inter-rater reliability statistics (as quadratic-weighted kappa) for the Summer Scoring dataset

We have run several experiments in the past in which it has been demonstrated that the inter-rater reliability of a scoring training set has a significant impact on the accuracy of the automated scoring model that is learned from that dataset. Therefore, it is recommended that some additional effort be put into future manual scoring sessions to increase the inter-rater reliability of LDC manual scoring. Higher agreement is typically achieved through close discussion of anchor papers with expert writing evaluation trainers.

## 5.5 Discussion for subsequent rounds of dataset collection

After our experience with the Summer Scoring datasets, some lessons were learned about improving subsequent rounds of data collection. This section collects, for the purpose of

archiving our thoughts, some informal comments that were made during discussions between SRI and LDC.

| Variable | Example values |
|---|---|
| **Writing type** | argumentative, informational, narrative, etc. |
| **Grade level** | 6-8. 9-12, etc. |
| **Subject area** | science, social science, ELA, etc. |
| **Topic** | [indefinite number of topics] |

There are four key variables that are important to consider when seeking to obtain an ideal training set for prompt-blind scoring: writing type, grade level, subject area, and topic (prompt).

The writing type and grade level variables are the most essential dimensions, because any time they change, the rubric for the core 5 traits changes. It is most important to get a good distribution across all combinations of these two variables. At an absolute minimum, we want at least 1 example in each cell of that 2 x 2 matrix (note that we didn't do any Grade 9-12 informational essays this summer.)

As for Variable 3 / subject area, my suggestion is that we try to get one of each of the subject areas in each cell of the 2x2 (Var1 by Var2) matrix. I realize that would result in 2 x 2 x 3 = 12 total prompts, which may be beyond what you are able to resource. So I guess we can call that "optimal"?

Lastly, if each of those 12 prompts covered a different topic, then I think we would be in a pretty good position on the topic variable, since the only goal there is just to get a wide variety of topics across the entire set.

I always feel confident saying "the greater the variety of prompts, the better we will do with the prompt-blind automated scoring" so long as the scoring is of a high quality. Therefore, I do think that having more than one prompt in each of the grade X style X subject cells would likely be a benefit. However, there are important caveats to consider in addition to the cost of additional scoring:

(1) There is a decreasing benefit to each additional prompt and/or data point (i.e., there are diminishing returns). I feel confident that we have never bumped up hard against a plateau in our previous experiments, where we've had no more than ~3000 total responses. I've suggested 2400 as a kind of "sweet spot" because (a) we are *still on a significant upward slope* at that point, meaning that none of the data collection effort will be in vain, and (2) we expect to be at an *acceptable* level of accuracy at that point as well. I estimate that collecting data beyond this sweet spot will probably increase our performance modestly, and is probably a requirement if we want to go from *acceptable* to *human level* accuracy. However, data for a 13th prompt is certainly not going to benefit us as much as the first 12 do.

(2) We prefer to collect at least some minimum amount of each prompt (~150) because (a) this improves the efficiency and consistency of the human scoring, which greatly influences the usefulness of the data and (2) so we can measure performance by-prompt in order to identify and correct for idiosyncrasies related to prompts. (Also because we put in some effort per prompt to gather up the background data.)

# 6 Modeling Additions

In this section we describe our progress toward enhancing the ATSE machine learning modeling system for the purpose of improving our ability to produce a highly customized prompt-blind LDC scoring model. We begin with a technical description of our newly developed machine learning workflow framework. We follow this with a summary of other related accomplishments in the area of modeling additions and improvements.

## 6.1 The workflow engine

The original incarnation of ATSE was an implementation in R of a model development architecture that applied multiple learners to multiple features extracted from multiple data inputs, and collected together the outputs of these learners into an ensemble of higher level features which were then fed to a suite of higher level learners. Recognizing the many ways to extract features, select learners, and combine outputs, we have re-architected the processing engine behind ATSE to enable a more flexible framework for training of models and scoring of new data sets. Our new *workflow engine* automates much of the human effort underlying the model development process. We are motivated to make the investment now, so that going forward we can accelerate a development process that is easy to configure, modify and extend.

The framework emphasizes re-use of processing modules, data products, and models. Its design simplifies the tasks of

- reconfiguring the flow of data
- adding new learners and processing steps
- iteration over parameters
- selecting the best model from a suite of competitors
- regenerating data

and automates their execution.

### 6.1.1 What is a workflow?

A workflow describes the movement of data through a processing pipeline, whether during model development, model evaluation, or scoring of new data. This flow can be represented as a directed graph. Nodes (graph vertices) represent transformations of data. Nodes are connected by any number of arcs (graph edges) that specify the flow of data in the form of inputs and outputs.

In our formulation, *data* is viewed abstractly as any information that can be stored in a file. This includes not only the traditional rectangular array of variables of arbitrary type, but also: files of unstructured data; associative arrays that map keys to values; and lists of parameters. Completed models that have been serialized for re-use can also be viewed as data.

Broadly speaking, a *node* represents any transformation that creates output data from input data. There is no minimum complexity for the processing encapsulated in the node. A node can be tasked with:

- generating features
- combining ('reducing') multiple data objects into one
- applying a previously developed model
- executing an external process
- selecting the optimum from a set of candidates

A workflow itself can be viewed as a node, and therefore can be embedded and invoked as a component in a larger workflow graph. This concept enables us to develop a library of processing pipelines and finished models that can be connected and reused in arbitrary ways. Rather than repeatedly re-implement the same logic for a particular data processing task, we define a standard workflow for that task, and store it in the library, for deployment as needed.

### 6.1.2 Specifying the workflow

We have devised a language to concisely specify a workflow and its processing steps using a graph-theoretic idiom. Processing nodes are defined with input and output ports for ingesting and emitting data. Arcs are defined to connect output ports to input ports. The ports themselves specify the name and type of the data.

Our framework flexibly handles not only *scalar* types (representing a single data type such as matrix or string) but also *hash* types. A data object with hash type is an associative array of items of the same scalar type, indexed by a set of keys. Processing steps that produce data products of hash type include:

- randomly splitting a set of records into folds; here the key is the fold index
- "exploding" text data into a tally of terms; the key is the term itself.
- scoring the same data set through a variety of models; the key is the model

In anticipation of the need to handle such data, our workflow language permits an arc to connect a hash output type to a scalar input type; this will be understood as an implicit request for the receiving node to *iterate* over the parameters. With this convention it is a simple matter to direct a workflow to carry out parameter sweeps, or perform cross-validation, or replicate the same action over multiple data sets. Our workflow engine can carry out this directive even if the number of parameters is not known in advance, e.g., if the number of keys generated by a node varies with the input supplied to the node.

### 6.1.3 Executing the workflow

Operationally, the workflow engine assembles the components of the workflow specification into a directed graph. Given a specification, the engine validates the internal consistency of the specification, constructs a representation of the workflow graph in memory, and determines the proper order of node execution demanded by the dependency on upstream data. As soon the required input data for a node becomes available, the node is placed into a processing queue, on which a team of workers (node executors) can operate in parallel. The interchange of data between the governing workflow engine and a node executor is handled via a simple API.

Each data object generated during workflow execution is written to disk and is registered in a master data base. We attach a *checksum* to each data set, which encodes its size and provenance. With this checksum we can automatically determine which data sets require re-calculation as a result of a modification to a workflow; and only those pieces downstream of the altered section of the workflow need to be regenerated.

The clean separation between a workflow and its execution facilitates the cycle of workflow modification and evaluation. We will maintain a repository of processing nodes, including entire workflows, along with the serialized models that they produce, for reuse as

components of more complex workflows. Workflows can be quickly assembled for both model development and scoring of new data.

Since the specification of a workflow is distinct from its execution, we are free to code the workflow engine in any language. Moreover, since the engine hands off the processing of each individual node to the node executor, the data transformation encapsulated in a node is hidden from the engine and can be implemented in any way. Individual nodes may be executed as calls to R routines, or implemented in Java, or executed by the operating system. As for the workflow engine itself, we avoid coding in the R language, which is too high level for an efficient production implementation. At this point we have coded a working prototype of the workflow engine in Python, and are in the process of porting this to Java.

We expect the workflow engine will greatly simplify the process of design and evaluation of competing model architectures and scoring of new data. It will permit us to accelerate the cycle of continuous experimentation and model optimization over a much larger universe of learners and hyper-parameters.

## 6.2 Model type additions

In parallel to development of the new workflow framework just described, we have also pursued the task of identifying and integrating specific candidate modeling approaches (model types) into our current ATSE system. One of our earlier achievements in this respect was an investigation and integration of the xgboost library. Having studied many machine learning discussion forums online, this approach seemed to be an excellent candidate, typically producing very accurate and generalizable models, and allowing data scientists to win many machine learning competitions. We completed a small exploratory experiment in which xgboost's performance was compared to our existing system, and the results were very positive. These results are documented in Section 9 below.

Subsequent to this experiment we discovered the 'caret' package as an important opportunity for rapid expansion of the set of first layer regression models in our model ensemble. Caret[1] is a set of functions that attempt to streamline the process for creating predictive models. The key benefit to ATSE improvement that caret provides is a uniform interface to more than

[1] 'Caret' (short for "Classification And REgression Training"); http://topepo.github.io/caret/

230 model types, which allows us to incorporate entirely new model types with very little effort. Since identifying this package, we have been able to do exploratory experiments similar to our xgboost study, this time investigating many more model types on our essay scoring prediction problem. It is also clear that the Caret interface fits elegantly into the workflow framework we have developed. Caret effectively turns "model type" into a tunable hyper-parameter that can be manipulated simply as a workflow variable.

While this is likely to be a great benefit to the accuracy of our prediction models, the opportunity significantly increases the risk of model overfitting. The more rapidly we can deploy and train models, the more likely we are to come across a good one *by chance*, which is likely to prove ineffective when generalizing to previously unseen data. We have therefore developed a more rigorous validation practice to control for this that hinges on the use of non-LDC datasets for the purpose of monitoring generalization and controlling overfitting. The method is described in detail below in the "Method" section of our quantitative evaluation in Section 9.

# 7 Prompt-blind Features

In this section we describe our progress toward implementing ATSE features that are specifically targeted to the LDC prompt-blind use case. In general, coding such features requires that the prompt itself, and any supporting material related to it, be studied by the machine so that the student response can be characterized *in relation to that material*. Subsequently, text analytic and natural language processing algorithms can be formulated to encode that characterization numerically.

## 7.1 Module data extraction

To develop methods for characterizing a response's relation to a prompt, it is obviously necessary to obtain thematic information about the prompt, and the teaching module the prompt is associated with. Fortunately, the LDC CoreTools infrastructure makes obtaining this information rather convenient, assuming programmatic access to that infrastructure. Simply by surveying the metadata and other materials associated with a module, one can find a significant amount of thematically related background material. For the LDC CoreTools system, this is typically

contained in the "Overview - Description," "Texts," and "Background for Students" section of the module, as well as Mini-Task handouts and resources.

Thus far, we have not been provided the programmatic access to LDC CoreTools that would be necessary to fully automate the process of extracting module data. However, this is not necessary currently since we are only investigating seven prompts and can perform the task manually. Indeed, that is our primary achievement to date. We have obtained access to the LDC CoreTools web site and manually downloaded all available thematically-relevant module background information. In many cases, this involves links to public resources outside the LDC Web site.

The procedure by which links to module resources were obtained started with the use of LDC CoreTools' "Export Module to PDF" feature. This feature was manually triggered for each of the modules, giving us one PDF document for each module containing all sections, including mini-tasks. Then, we created a short python script that uses the *PyPDF2* library to automatically extract all link URLs from each PDF document. The resulting list of extracted links, by module, is provided in the Supplement A spreadsheet under the MODULE_LINKS sheet. We then used the *lynx* program to automatically retrieve the textual contents, if any, present in the referenced resources. In some cases, the links referred to written documents with topically relevant text. In others, the links referred to resources like videos, or dead URLs. Therefore, it was necessary that we apply a post-processing step that we refer to as "prose extraction," which automatically identifies and extracts only the portions of the documents that are similar to English sentences (as opposed to tables, ads, boilerplate, etc.). Only the "prose" portions of the documents are used to train prompt-relational features. Some examples of the extracted prose are provided as a supplement to this report.

## 7.2 Public resource extraction

As our algorithms are statistical in nature and improve with the availability of larger amounts of textual data, we established a sub-goal in this phase of the project to develop a fully automated method for using LDC module data as a "seed" for conducting a more extensive data collection against public resources on the web.

The procedure works as follows. First, we obtain the thematically-relevant module data from the LDC CoreTools system as described above. This gives us an initial picture of the

prompt domain and subject matter, and a means for determining the words and phrases that are most relevant to the prompt via a measure such as TD-IDF. This then gives us a list of key domain terms that can be used to conduct a web-based public data search, e.g., Bing or Google. For example, the term "water cycle" would be identified as a seed term for the Summer Scoring WATER module. A query of this term in Google would return a list of public documents relevant to those query terms. The response from Google is then analyzed, and the links returned are downloaded and used to further expand the set of documents we have available, creating an even more thorough dataset for the system to analyze. All of this translates to more accurate statistical measures of association between student responses and the prompt, which is the underlying technical goal of this task.

## 7.3 Prompt-relational feature implementation

The module data extraction and public resource extraction process described above produces a corpus of topically-relevant texts for each prompt. Obtaining these corpora has two main purposes, which we refer to as *meaning representation* and *meaning relevance*, each of which is associated with a type of implemented prompt-relational feature described below.

### 7.3.1 Learning domain meaning representations

The first reason for obtaining a topical corpus for each prompt is to learn adequately nuanced meaning-based representations of the vocabulary of words and phrases in the domain of the prompt. In many domains, there are words and phrases that do not occur in a general corpus of English. For example, it is entirely possible, even in a corpus of one million randomly sampled documents, that a phrase like "Red Scare" (a key phrase in one of the Summer Scoring prompts) may not occur at all or may occur only very rarely. Because the system relies upon the contextual analysis of many occurrences of words and phrases to understand their meanings, if a word or phrase only occurs a few times, an adequately nuanced understanding cannot be obtained.

By supplementing the background corpus with topically-relevant documents obtained as above, the system is provided an opportunity to learn a meaning representation for these words. This then allows the system to better understand the words and phrases in the student responses, which of course will tend to be in the domain (for good responses at least).

To implement the "meaning representation" prompt-relational features is actually rather straightforward. All that is required is that these corpora are added to the background corpus that is already used in the modeling process. No new code is actually required other than the software used to obtain the prompt-relevant supporting documents themselves, which we described in the previous subsections.

#### 7.3.2 Identifying domain meaning relevance

The second reason for obtaining a topic corpus for each prompt is so that meaningful relationships between the response and the prompt can be measured. In the absence of prompt material providing context, the system has no idea whether the text the student has submitted is relevant to the prompt. In the prompt-specific training scenario, this was not required, since the response could be compared to other high-scoring responses in the training set to see if the subject matter was similar to those responses. However, in a prompt-blind setting, we have no assurance that some other high-scoring example in the training set is a response to the same prompt, since the training set contains responses from multiple prompts.

To implement features that measure domain meaning relevance, we use a number of techniques, all of which are built around our core method for assessing meaning-based associations between two texts, which relies upon both weighted counting of words and phrases, as well as collapsed representations of sentences and larger portions of text. Specifically, for example, we start with a non-prompt-relational feature such as a count of the number of times a word occurs in the response. To make this a prompt-relational feature, we can weight it by the relative frequency of that word in the domain corpus, as compared to a general background corpus. Similar weightings can be done for other previously computed features that go beyond individual words and phrases, such as the features which aggregate entire sentences into a single vector-based representation.

# 8 Transformations

## 8.1 Pre-processing with lossless transformations

Before presenting features to a learner, it is possible to pre-process the data with a lossless transformation. Such transformations include:

- *Standardization*, also known as *z-scaling*, in which the sample mean is subtracted from each item in a list of observations and the result is divided by the sample standard deviation. The resulting modified feature will have a sample mean of zero and a standard deviation of 1.
- *Normalization*, where each value is divided by the maximum observed value. Optionally, prior to this division, the values may be centered through subtraction of the feature mean.
- *Percentile transformation*, such that each observation in a list is replaced by its percentile rank. A transformation of ordered categorical data known as *ridit scaling* is a form of percentile transformation.

The benefit of these normalizing transformations is that all features are placed on a common, unit-free, scale. By doing so, the transformed features can be directly compared and also combined, and it is readily apparent which values are extreme versus representative.

In classical linear regression it is desirable to perform some form of normalization prior to parameter fitting, which not only yields scale-independent regression coefficients, but also reduces numerical instability in calculation of those coefficients. Modern learners, on the other hand, are typically impervious to changes in scale of the features — their numerical stability and learning ability is the same whether or not these transformations are performed.

Since we expected little difference in behavior from using this approach with the high-performing modern learners added to our system during Subtask 1.2, we declined to expend effort on experimenting with normalizing transformations.

## 8.2 Dimensionality reduction of the feature set

*Principal component analysis* (PCA) is a well-known technique for reducing the dimensionality of a data set. It is a statistical procedure that finds a set of linear combinations of the features that will maximally account for the variability in the data. These linear combinations, called principal components, are ordered so that each succeeding component has the highest variance possible under the constraint of being orthogonal to the preceding components. In a PCA one typically selects only the top principal components, enough to account for a large percentage of the variability, thereby achieving a reduction, often considerable, in the dimensionality of the feature set. Since the principal components take the place of the original feature set, such a data transformation is lossy.

We used the caret framework to test the efficacy of PCA for each of the selected learners. The caret package integrates a PCA option into its cross-validation capability. We requested that caret pre-process the training feature set to feed only the top principal components to the learner, with the default behavior of performing z-scaling of each feature prior to PCA, and selecting enough components to account for 95% of the variance.

Unfortunately, our quantitative evaluation results indicated that the loss of information caused by PCA is more detrimental to the learning process than the transformation is beneficial. Instead of making the prediction problem easier for the learner, the approach generally degrades the learner's performance. **This degradation was apparent for all of the high-performing model types, and no improvements were achieved using this strategy.**

However, one advantage was observed. The use of PCA causes a dramatic reduction in computation time, afforded by the elimination from consideration of features that carry little predictive content. This is true in particular for the random forest learners, which completed processing in minutes rather than hours.

## 8.3 Predicting a categorical target

Most learners are equipped to do both *regression* (to predict a numeric target) and *classification* (to predict a categorical target). Thus far in this project we have used each learner in regression mode, presenting the score to be predicted as a numeric target; the predictive model thus learned will return a floating point prediction. An alternative to regression is classification, where each possible target value is a separate category to be learned; for each observation (essay) the predictive model will return a score from among the set of possible categories.

Again we conducted our modeling study in caret using the top performing learners. In the caret framework it is a simple matter to request a learner to perform classification instead of regression: we simply change the type of the target from numeric to categorical. (This is achieved by requesting in R that the target be interpreted as a *factor* object.) One subtlety of the target distribution led to frequent model failure: certain target values are recorded as the average of two or more scores. Passing these to the learner as valid categories to be predicted would lead to complaints about the infrequency of those categories, and a failure of the algorithm. The solution is to round the target values to those allowed by the scoring rubric before proceeding with learning. In doing so there is a loss of target information compared to the regression

situation. On the other hand, this rounding guarantees that only valid scores will be learned--the predicted score returned in classification will always be admissible, whereas in the regression use case the output is a floating point prediction which must be rounded to the nearest conforming value.

As with our evaluation of the dimensionality reduction strategy, our quantitative evaluation results for this strategy also indicated that the loss of information is more detrimental to the learning process than beneficial. **Again, this degradation was apparent for all of the high-performing model types, and no improvements were achieved using this approach.**

For predicting a target such as essay score, it seems the classification approach is disadvantaged by the loss of information in converting a numeric outcome to a category: the classifier cannot take advantage of the natural ordering embodied within a numeric score, and thus we can expect some loss of predictive accuracy compared to regression. (With some learners it is possible to take advantage of this ordering, by declaring the type of the target to be an *ordered factor*.)

## 8.4 Predicting a binary threshold target

Another approach to prediction of a numeric target is to combine the predictions of a family of sub-models, each sub-model predicting a different *binary threshold* target. For example, for a score taking integer values from 1 to 4 one can define four sub-models: the first predicts the probability that the score is greater than or equal to 1, the second predicts P(Score $\geq$ 2), the third P(Score $\geq$ 3), and the fourth P(Score $\geq$ 4). (Since the first of these models will yield identically 1, there are only three binary targets to learn.) These four model outputs can be combined into a prediction by using the well-known formula for the expectation of a non-negative random variable in terms of the tail probabilities of the random variable:

$$E(X) = \int_0^\infty P(X \geq t)dt \qquad (1)$$

In the above example of integer-valued scores from 1 to 4, this formula simplifies to

$$E(Score) = 1 + P(Score \geq 2) + P(Score \geq 3) + P(Score \geq 4) \qquad (2)$$

and in general for predicting a discrete numeric target such as an essay score it is enough to build binary predictors that capture every possible score threshold, and analytically combine the observed predictions using (1).

We employed the caret cross-validation framework to study this approach. For each learner we requested a number of *binary* classification models to be built, with the output being the probability of the 'positive' class. The predicted probabilities obtained from cross-validation are then combined (outside of caret) using an analytic calculation similar to (2).

It was with this approach that we were finally able to achieve a positive performance outcome. **For both the xgbTree and xgbLinear model types, the strategy induced a consistent improvement in scores, most notably for the xgbLinear model type. For xgbLinear, using the binary threshold strategy caused a mean 7.9% improvement across all five traits. For xbgTree, the improvement was a modest but positive mean of 1.3%.**

There are a number of possible reasons the 'integration of binary thresholds' approach did not work for the other model types. At the lowest score thresholds, the preponderance of target values for these binary models are positive examples, while as the threshold increases, the negative examples will dominate. In other words, at both ends of the score distribution the ratio between positive and negative examples is skewed. This leads to instability in learning the binary target. If training is not conducted with care, the resulting model (if it converges at all) may suffer from overfitting, or may learn a trivial result (always returning the majority class). Here too, the natural ordering between scores is lost in the transition to binary threshold models. There is no enforcement of the requirement that the higher the score, the less likely the score; consequently, it is possible to observe non-monotonic behavior in the predicted probabilities as the score threshold increases.

## 8.5 Discussion

Despite the relatively underwhelming overall performance of these approaches, there were isolated but clear improvements that can be used help to make the system more accurate for some traits. Also, the by-products of these approaches can play a more general role in the modeling enterprise — as augmentations to the feature set. We can adjoin the top principal components to the feature list instead of replacing them entirely. Or we can build binary threshold models as a preliminary task and regard the predicted probabilities calculated by these

models as a new class of features. Indeed, the model outputs from all of the approaches studied here can serve as additional *higher-level* features; this is the approach that we took in the original incarnation of the essay scoring engine. It is also an approach that we will explore later in the project as time permits.

# 9 Quantitative Evaluation

This section presents the results of two system evaluations: (1) a baseline evaluation to determine the scoring accuracy of the original version of ATSE in relation to LDC datasets and the LDC prompt-blind scoring use case, and (2) an interim evaluation conducted at the completion of Subtasks 1.1 and 1.2 to determine how much the system's accuracy has improved as a result of introducing the feature extraction and modeling approaches described above. We present these two evaluations in the subsections below, but begin first with a description of the evaluation methodology.

## 9.1 Evaluation method

For the LDC use case, our principal goal in evaluating ATSE is to measure how accurately it can score responses to previously unseen LDC prompts. This is the definition of the "prompt-blind" use case.

#### Prompt-folded cross-validation and generalization

ATSE is a supervised machine learning system. The scoring models that ATSE learns are built by observing a training dataset of manually scored texts. Because of this, it is inappropriate to evaluate ATSE's performance using examples from the training set. Doing so can be viewed as a form of "cheating," such that it prevents a true measurement of the primary aspect of the system's accuracy that we are trying to evaluate — it's ability *generalize* to new (unseen) data. Without measuring generalization, there is a significant risk of producing a model that performs well on the training data, but which performs poorly on other datasets, otherwise known as overfitting.

A more appropriate and commonly used method of model validation used in machine learning is called "k-fold cross-validation."[2] In k-fold cross-validation, the training dataset it split

[2] https://en.wikipedia.org/wiki/Cross-validation_(statistics)#k-fold_cross-validation

(usually randomly) into multiple partitions or "folds." Then, the system is trained using all but one of these folds (called the "holdout" fold), and then evaluated based upon its prediction performance on the holdout. This process is repeated for each fold, and the mean of the multiple results is then reported.

In the LDC use case, however, it would be inappropriate to perform a naive *random* k-fold cross-validation experiment. This is because a randomized partitioning would result in responses from any given prompt appearing in both the training and validation set.

Instead, it is necessary that we force the fold partitions to correspond to a partitioning of the training data by prompt. We will refer to this as "prompt-folded cross-validation." In the case of the Summer Scoring dataset specifically, we have 7 prompts. This results in each round of validation having a training set composed of responses to 6 prompts, and a validation set composed only of a holdout 7th prompt. We then conduct 7 rounds of validation and tabulate those results.

We note that this type of evaluation, given our training set of 7 prompts, actually produces an estimate of prompt-blind scoring accuracy that is based upon a 6-prompt training set. Training the system on all 7 training set prompts would likely result in greater accuracy, but we do not have the means of empirically assessing that accuracy until another new hand-scored prompt dataset is available.

We also note that there is an additional risk of overfitting, i.e., a lack of generalization, even if cross validation is used to select models with what appears to be good generalization. This risk comes from the customization process as a whole. If the actions we take to improve the ATSE framework, such as feature development and modeling structure configuration, are driven by our observation of the cross-validation error, then the customization process itself is lacking a cross-check against a higher level of overfitting. To address this concern, the ideal scenario would be to select one or two prompts and reserve them as a long-term customization process holdout. Our customization process would then proceed, using the non-held-out training data and cross-validation procedures to guide us, and only at the end of each major round of customization (e.g., every couple of months) would we evaluate how the system performs on the long-term holdout. This is the ideal simulation of the true LDC use case and will give us the best overall estimate of accuracy and generalization. However, use of this additional procedural check

for generalization can only really be effective once we have more than 7 prompts. We therefore postpone use of this approach until the next round of hand-scored data becomes available.

## 9.2 Baseline evaluation

The goal of the baseline evaluation is to determine the scoring accuracy of the original version of ATSE for LDC datasets and the LDC prompt-blind scoring use case, so that a relative assessment can be made of our future ATSE improvements. To do this, we perform the prompt-folded cross-validation experiment described above, using an unmodified version of ATSE.

The results of the baseline experiment are documented in detail in the matrices in the sheet named BASELINE in the Supplement A spreadsheet. The top-most matrix in that sheet shows the inter-rater reliability by prompt and by trait. **The mean inter-rater agreement, expressed as quadratic-weighted kappa ("qkappa" henceforth), across all traits and prompts is 62.8%.** This number expresses a summary of our target accuracy for the system.

The next four matrices in the sheet present both the prompt-blind and prompt-specific results of the baseline ATSE system as qkappa by prompt and by trait. Two of these matrices show the results as *absolute* values, while the other two show results as *relative* to inter-rater reliability. Measuring accuracy as relative to human agreement allows us to conveniently consider 100% as our target accuracy.

We note that previously reported prompt-specific results are slightly different than the current results (see the grey cells). The reason for this is the inherent non-determinism in the experiment, where each time a training run is executed, a slightly different model is produced.

**To summarize the results of the baseline experiment: the system achieves a mean baseline human-relative qkappa performance of 51.3%. This is our starting point for the project, with our goal being to increase this to 100%, so that the system is effectively producing scores in a manner that is as accurate as human scorers.**

## 9.3 Interim comparative evaluation

The goal of the interim evaluation is to determine how much the system's accuracy has improved as a result of introducing the modeling approaches and prompt-blind features developed since the beginning of the project. We describe the results in three stages. In the first stage, we present an exploratory study of caret model types. In the second stage, we present improvements that result

from the application of a chosen set of these new caret-based modeling approaches (introduced above in Section 6.2). In the third stage, we describe the additional improvements that arise from the inclusion of new prompt-relative features (introduced above in Section 7.3).

### Caret model exploration

One of the key approaches that was planned for Subtask 1.1 was to explore the addition of promising model types to the ATSE modeling ensemble. Initially, the task was begun by studying *xgboost*, but we subsequently discovered the *caret* package, which allowed us to rapidly accelerate the number of model types that we could explore (including xgboost). Caret is an R package that provides a common interface to many model types, importantly including built-in routines for hyper-parameter tuning by cross-validation.[3]

As a first step to using the caret package, we extracted a small subset of available ATSE features from the LDC prompt-blind experiment, selected only one trait for scoring, and manually studied how well every available caret model performed. We selected a small number of features instead of using all of them because we expected that many of the caret model types would not scale well to our very large set of features. Instead, our goal in this first step was primarily to weed out those caret models that are clearly low performers and not adaptable to our use case.

The results of our exploratory caret model study are documented in detail in the Supplement A spreadsheet on the sheet named CARET_MANUAL. In the sheet, we have listed the 134 caret-accessible models that are available for performing regression. Additionally, we have noted, when the model was successful, the attained qkappa value and amount of time required for processing in seconds. Also, we have attached comments and a record of errors for later study and debugging. Many of the models produced errors during training. Many others were manually skipped because errors were anticipated (such as when it has already been discovered that a model's supporting package will fail). Still others were run, but then

---

[3] We refer the reader to a paper by Fernandez-Delgado, et al., called "Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?" in Journal of Machine Learning Research 15 (2014) 3133-3181. The paper presents an extremely thorough study of many model types across many datasets for the problem of classification (as opposed to regression). Many of the best performing models are those provided by the caret package.

prematurely manually terminated because the processing time was determined to be excessive. Table 3 shows the frequency of different outcomes achieved for the 134 models studied.

Table 4. Counts of the various outcomes in our exploration of available caret regression model types

| Result | Description | Count |
|---|---|---|
| **bad** | The model finished training, but the results were essentially random. | 6 |
| **crash** | The model crashed the entire R system while training. | 1 |
| **error** | An error occurred while training and no results were produced. | 25 |
| **good** | A reasonable result was achieved. | 39 |
| **hang** | The model caused the R system to become unresponsive. | 4 |
| **long** | The model took too long to train and was manually terminated | 13 |
| **skipped** | The model was skipped because an error was anticipated. | 46 |

As the table suggests, 39 models completed training with a positive outcome. There were an additional 6 models that completed, but which produced essentially random outcomes. (We note in the spreadsheet that we would like to continue to assess these 6 models in the future, as it has been reported that they often work well.) The 39 successfully trained models were chosen for use in the next phase of our experiment.

### Addition of good caret models, with original features

In the next phase of our experiment we applied the 39 successfully trained caret model types to the full set of original ATSE features and all 5 traits in the LDC rubric. The results are documented in detail in the two matrices at the top of the sheet named CARET_MODELS in the Supplement A spreadsheet. This matrix shows the prompt-folded mean qkappa validation result for all 39 model types. The models are ranked by their mean qkappa across all five prompts, with the best performers at the bottom (blue indicates better and red indicates worse). The matrix to the left shows the raw qkappa scores, while the matrix on the right expresses the scores as relative to the results achieved in the baseline experiment described above in Section 9.2.

**Four of the models achieved at least a 20% improvement over our baseline experiment, while 15 models achieved at least a 10% improvement. By selecting the best**

**performing model for each trait (see the row labeled "MAX"), the mean improvement over the baseline model can be raised to 37.3% (see the highlighted yellow cell).**

### Addition of prompt-relative features

In the final phase of our experiment, we added our new prompt-relative features, described in Section 7.3, to the existing basic ATSE features and measured the improvements. The results are documented in detail in the three matrices at the bottom of the sheet named CARET_MODELS in the Supplement A spreadsheet. The matrices show the prompt-folded mean qkappa validation result for a selected 6 model types, chosen based upon their good performance in the previous stage of the evaluation, as well as their ability to do training in a reasonable amount of time.

There are some caret models, particularly the random forest based models (e.g., *rf, Rborist*, and *parRF*), that have been documented as being highly accurate models. However, due to time constraints on completing the experiment, and poor compute speed for these algorithms, final results were not obtained in time for this report. In the future, when time for training is less limited, we plan for these models to be added to the ensemble.

**Four of the six selected good models achieved at least a 10% improvement over the results of the caret experiment with only original features. By selecting the best performing model for each trait (see the row labeled "MAX"), the mean improvement over the selected best models for the original feature experiment was 5.1% (see the highlighted yellow cell).**

### Summary

**Our additions to the model types and feature extractors used in ATSE has resulted in an approximately 44.3% improvement over the original ATSE system as it was at the beginning of the project. This equates to a 70.6% performance relative to human inter-annotator agreement. While we have achieved a substantial improvement over the baseline approach (51.3% relative to human agreement), the need for further improvement remains before the system can be assessed as statistically similar to human grading.**

# 10 ATSE REST / Microservice API

While not part of the current project, we would like to report on an important development at SRI with respect to the ATSE framework.

SRI's Education Division are world-class experts in the area of educational assessment and conduct a significant volume of assessment research each year. This research often includes the deployment of assessments in schools, which can be costly without the right infrastructure. Our Education Division recently identified the problem of reducing the effort and cost of deploying assessments as a key challenge and allocated some internal research and development funding to pursue development of a SRI-accessible instantiation of the open-source Tao assessment platform for the purpose of facilitating SRI's assessment research.

As part of this internal SRI-funded project, the ATSE team was asked to develop an interface for ATSE that would allow it to be conveniently linked with the Tao platform, to allow for the Tao platform to include automated assessment of the SRI-developed ECS "Exploring Computer Science" assessments (which we have been developing ATSE scoring models for).[4] This development project was very recently started in October and is being performed by a dedicated full-time software engineer. It is expected that the task will be complete by the end of 2016.

What this means for LDC is that we expect to have an easily accessible solution for deploying the ATSE software to the LDCs CoreTools environment much sooner than originally anticipated. ATSE will have a convenient, well-documented, Web-accessible REST interface, which could potentially be integrated into LDC's solution without any further effort on SRI's part other than documentation and support. We think this is an exciting development and hope LDC are pleased to know that we are a significant step closer to the final objective of full integration of ATSE into the LDC platform, even though this was not part of our current LDC-funded project.

[4] https://csforallteachers.org/blog/release-of-the-ecs-assessments-cumulative-units-1-4-assessments-available-now

# 11 Progress summary archive

This section archives “Progress summary” sections from previously delivered progress reports, for the purposes of reviewing project progress over time.

## 11.1 Progress summary from Interim Report #1

We begin with a short summary of our progress with respect to these targets to-date, reserving a detailed description for the sections below.

Our first major accomplishment in the project has been the completion of a baseline ATSE evaluation using a preliminary LDC dataset (henceforth, the "Summer Scoring" dataset). This has set an important foundation for our work, giving us the necessary baseline measurements to comparatively evaluate our progress in improving scoring accuracy going forward, and getting us familiarized with using and analyzing LDC-collected datasets and responses to LDC prompts. After completing the evaluation, we provided documentation of the results to for use in a Department of Education proposal, submitting a rough report in August 2016. *An extended version of that report on the Summer Scoring dataset is provided in Section 0 of this document, while a description of the baseline evaluation is described in Section 9.*

Our second major accomplishment has been the design and development of a novel machine learning workflow framework. The framework addresses one of the key challenges associated with our goal to introduce new models into the ATSE system. Recognizing that the optimization of ATSE's modeling components for the LDC use case is likely to be an ongoing task not isolated to the first months of this project, and given delays in obtaining high quality training data, we decided to invest effort early on to maximize later benefit. We decided to re-architect the ATSE modeling engine to enable a more flexible method for formulating and training models. Our new workflow engine automates much of the human effort underlying the model customization process. Additionally, we have come upon, and have subsequently integrated into ATSE the "caret" package, which is an R-based API for conveniently training hundreds of different model types, including our targeted "xgboost" library. *This modeling improvement work is described in detail in Section 6 below.*

Finally, with respect to our other main interim goal of prompt-blind feature development, our progress has unfortunately not kept up with originally planned pacing. While we have managed to instantiate the mechanisms for extracting relevant background reading materials

manually, and we have implemented a new automated method for automatically retrieving publicly-available related materials via Web search queries, we have not yet completed what we feel to be a comprehensive set of prompt-relational features. We therefore suggest that execution of this task be extended for another month in order to achieve the goals we set for it. *Documentation of what we have achieved thus far on this task is provided in Section 7.*

Despite our effort being partially delayed, we think it is important to proceed with documenting our progress to-date. The current report is therefore missing one important element that was originally planned for inclusion — results of a comparative quantitative evaluation showing how our developments have improved scoring accuracy. These experiments are underway, but more time is needed to finish them, analyze their results, and document them. Our new plan is to complete these experiments and produce an updated version of this document ***by November 18th***. In the meantime, we hope to receive LDC's comments, questions, or requests for further detail on what is present in the report, the responses to which we can also include in the subsequent version.

## 11.2 Progress summary from Interim Report #2

Our goal in Subtask 1.3 was to complete an investigation into multiple strategies for extracting maximum predictive performance from ATSE, specifically focusing on strategies that were complementary to implementation of new features or machine learning algorithms (since these approaches were reserved for other tasks). Instead, this task would focus on the isolated challenge of manipulating feature data that was already being generated, in a way that would benefit the learners already in our toolbox.

The completion of Subtasks 1.1 and 1.2 and the integration of the caret framework gave us new information about the nature and performance characteristics of several component model types, and one of the first determinations we made for Subtask 1.3 was that some of the strategies we originally proposed were unlikely to improve performance when using these new model types. This is because many of the best performing model types already have inherent capabilities that encapsulate our proposed transformations. Also, given the large gains achieved with Subtask 1.2, our expectations for the amount of performance gain that would be achieved were diminished, since those improvements would likely overshadow improvements obtained via transformations that do not explicitly add information to the data.

Adjusting our approach, we decided to focus instead on strategies that were hypothesized to be *complementary* to the high-performing modeling types introduced in Subtask 1.2. We decided to focus on three main themes: (1) pre-processing of the feature set with lossless transformations, (2) dimensionality reduction of the feature set, and (3) alternative targets for prediction.

Each of these three avenues of research is discussed in a separate section below. In each case we carried out a quantitative evaluation similar to the one described in Section 6.3 our previous Interim Report #1, selecting the top performing learning algorithms identified in that report (i.e., glmnet, Rborist, xgbTree, parRF, xgbLinear, rf, gcvEarth, earth, svmLinear3, and gbm), applying our selected transformation, and then comparing the performance result to that obtained without using the transformation.

In summary, the performance improvements we obtained in Subtask 1.3 with our transformation approaches were not as high as originally anticipated. Only one of our strategies (predicting a binary threshold target, see Section 8.4 below) was able to consistently improve the outcome for any particular high-performing model type. **One key highlight was that this strategy garnered a mean 8% improvement for the xgbLinear model across all five scoring traits. However, because xgbLinear is not our *best* performing model type, our improvement over the *best* model type for each trait averaged only 1.7%, with 3.9% achieved for the "Conventions" trait and 4.1% for "Organization." Improvement on the other traits was, unfortunately, insignificant.** This result does not meet our 5-10% improvement objective and suggests that more headway can be made by shifting focus back to feature development — the subject of the next task in our project schedule, Subtask 1.4.

In the following sections, we present the technical details of the transformation strategies we explored in Subtask 1.3.