Project 1 for CS421 - University of Illinois at Chicago
Peter Hanula   phanul2(at)uic.edu
Matthew Healy healy5(at)uic.edu

--->SETUP<--
The text files containing the essays to be graded should be placed in the directory 'input/test/original. All essays in this folder will be graded in a single run of the program

Once the program files have been placed the program can be run by directly by using the file AutoGrader.jar located in the main directory. The program can also be run from the command line using the command 'java –jar AutoGrader.jar. Code was designed to run on Java 8. When program has finished running a message indicating that it is done will be printed.

Output from the program can be found in the 'output' directory in the file 'result.txt'. Only the results from the most recent run of the application will be saved in this file.

--->TECHNIQUE<--

POS tagging was used to identify errors in subject-verb agreement, as well as to identify missing or extra verbs in the sentence. We identified cases from the training essays in which such errors occurred and made note of them. We then identified the sequence of tags assigned to these erroneous phrases. We now have the program search for these sequences of tags that we know indicate grammatical errors.

E.g. The tag sequence MD -> VBP is and error. A modal verb should not generally be followed by a third person singular verb.
Also the tag sequence VBP -> VBP should result in an error. Two third person singular verbs should not come in sequence.

The OpenNLP parser was used to assist in checking for sentence well-formedness. Common errors were checked for by comparing known errors to parse results.

Topic coherence was checked with the aid of WordNet. An array of words related to the topic was created. This list was then checked against the nouns in the essay. The percentage of nouns that were considered related was used to produce the final score for topic coherence.

--->REPORT<---

Everything in the project worked to some degree, but there were many areas that did not work as well as we would have liked. The spelling checker works pretty well. We use two different spellcheckers for a couple reasons. The biggest reason is that the OpenNLP lucene spellchecker will not check words under three letters long. This is a problem because it is entirely possible that short words could be misspelled. So, we have the lucene spellchecker ignoring short words and the results of both spellcheckers are averaged to get a final spelling score.

The subject-verb agreement is sound in theory, but could use some more work to improve. We currently check for commonly appearing error sequences. If any of these POS sequences occur an error is reported. The problem here is that we have no doubt missed some sequences that should be flagged as erroneous. Also, some of our cases feature rules that are only erroneous most of the time. Given more time and manpower, we could construct a more complete and correct list of rules that could correctly identify more errors.

The verb-tense/missing/extra verb case is another spot where the theory is sound, but more time and work would be needed to correctly identify all relevant rules. The algorithm used to check for missing and extra verbs is pretty naïve and based on a small set of rules. With more time, and more essays, we could more easily identify erroneous states and come up with rules that would allow these errors to be identified quickly and accurately.

The sentence formation checks also would require some work to improve the system. Currently, the program checks for very basic error sequences in the parses for the sentences to determine if a sentence is appropriately formed. Again, this process could be improved by finding more and better rules to determine if an error is present in the text. Presently, we only check against the most simple and common errors, but surely there are many more errors that could be identified at this stage, given more rules.

The sentence counter for the essays works pretty well, but it could also use some work. Ideally, this counter could be retrained on a corpus of essays of this type to yield more accurate results. There are many different types of errors in sentence separation that students make in these essays that are strange and hard to detect. Periods are sometimes used, but no spaces follow them. This was an error we discovered while working on part II. Our sentence splitter would not count sentences if there was no space after the period. This problem has since been fixed. Sometimes periods are used after something that is not a sentence. Sometimes no periods are used over what should be several sentences. In all, it makes it fairly difficult to identify how many sentences are present. Again, ideally, this could be helped by training a sentence detector on a corpus of similar essays.

The coherence categories is where this program could use the most improvement, and where it would need the most improvement to make a truly next-level product. Currently the algorithms used to check for essay coherence are pretty naïve. The presently available tools don't work as well as we would like. The Stanford coreference detector does not seem to be especially accurate, so the project is limited by that. If a more accurate coreference detector could be found that would likely improve the quality of the product. Also, we only check for pronoun reference resolution. Adding more checks for coherence could greatly improve the accuracy with which we could grade the essays. It would, of course, also make the project much more complex. Even within pronoun resolution we make several simplifying assumptions, such as the assumption that if a pronoun doesn't have an antecedent in the last two sentences, then it doesn't have one at all. Reducing the number of assumptions and increasing the breadth of the coherence checking could greatly improve the performance of the program, but it would also make it much more complex.

Finally, the topic coherence checking also works to some degree, but it could use some improvements. The functioning part of the algorithm generates a list of words related to the topic. This is then checked against the nouns in the essays. The more matches are found, the more likely the essay is to be on topic. This, however, is not a perfect algorithm. First, the list of related words is by no means complete. Second, the algorithm may score creative essays very poorly because they do not use as many of the expected words as some other, less well-written essays. Finally, there is not guarantee that just because words from the list are used that the essay is not straying away from the topic to something that is tangentially related.

Overall, the biggest thing we learned from working on this project is that natural language processing is very difficult. The tools available are limited in their accuracy, especially when unexpected errors occur in the essays. There are many different ways in which errors can be found, and creating rules representing them all is very difficult and time-consuming. Apart from this, we learned how to use some of the many available tools to perform a real-world function. We also learned how to identify ways to interpret data received from these tools and augment it to give more meaningful results.