Automatic Essay Grader

Project 1 for CS421 Project Report – University of Illinois at Chicago

Sandeep Joshi: sjoshi37@uic.edu
Sakshi Panday: spanda7@uic.edu

a) Length of the Essay

Length of the essay is being calculated by counting the number of sentences. As there could be punctuation and capitalization mistakes, we try to get the sentences using parse tree of the raw data.
We take essay as an input and create a parse tree using Stanford CoreNLP annotation. Once, we have the parse tree, we calculate the number of sentences based on start tag 'S' and few other conditions.

The primary type of relevant example of parse tree structures are:

1. *"I run I jump"*

```
(ROOT
 (S
  (NP (PRP I))
  (VP (VBP run)
   (S
    (NP (PRP I))
    (VP (VB jump)))))))
```

Giving us **two** sentences as count.

2. *"I run, I jump"*

```
(ROOT
 (S
  (NP (PRP I))
  (VP (VBP run) (, ,)
   (SBAR
    (S
     (NP (PRP I))
     (VP (VBP jump)))))))
```

Although, we can see two 'S' tags here, but as the preceding tag for the second 'S' is 'SBAR' i.e. subordinating conjunction, it is counted as only **one** sentence.

3. *"I run and I jump"*

```
(ROOT
 (NP
  (S
   (S
    (NP (PRP I))
    (VP (VBP run)))
   (CC and)
   (S
```

(NP (PRP I))
(VP (VBP jump))))))

Although, there are three 'S' here, but other two are enclosed in first 'S' as they are joined by a conjunction. Hence, we count it as **one** sentence.

b) Spelling Mistakes

Spelling mistakes are recognized using the autocorrect package. Once we recognize that there is a spelling mistake in the word, we proceed to further investigate the kind of spelling error. All spelling error are not equal. For example, a mistake like 'calender' is slightly more egregious than a mistake like 'embarassing' which has a missing character which was repeated in the correct word. Another benign mistake is confusion of 'ie' over 'ei'. For example, 'recieve' is a benign mistake over something like 'argoment'. We identified few categories of such errors and weighted them accordingly. The categories we considered are -

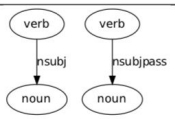
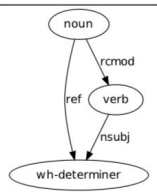
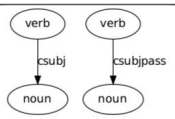
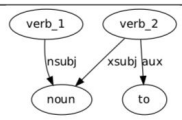
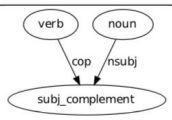
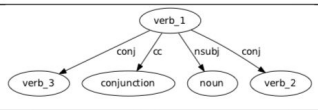
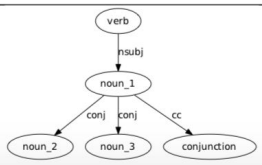
Error Pattern	Pattern Description	Error weight
Ie -ei confusion	Confusing over use of ie over ei and vice versa	0.5
Repeated character mistake	Missing character is repeated in the correct word	0.65
Vowel replacement mistake	Confusing one vowel for another	0.9
Consonant replacement mistake	Confusing one consonant for another	1
>2 characters difference in incorrect and correct word	Correct and incorrect words differ heavily	1.5

Further we tried to consider homophones. If the phonetics of the incorrect word the correct are same, then the incorrect word does not need to be penalized. But the package pronouncing was giving more false positives and very less accuracy. We have dropped it for now in part 1. Further we are considering not only to just check the phonetics but also to investigate if calculating cosine distance between correct and incorrect words will give us any more insight or better accuracy.

c) Syntax/Grammar

i) Subject-Verb agreement

Subject verb agreement accuracy requires to find the subject and the verb of the sentence. Subject does not always occur beside the verb. Also, dependencies like nsubj does not always provide the dependency between noun and verb. This misses few subject verb pairs which would have not agreed. A paper we investigated provided rules to apply on the dependencies to extract subject and verb. Below are the rules mentioned in the paper[6].

	Description	Path
1	Direct subject	
2	Path through Wh-determiner	
3	Clausal subject	
4	External subject	
5	Path through copula	
6	Subject in a different clause	
7	Multiple subjects	

We are using the rules 1,3 and 5. Implementing these gave good results in terms of identifying the subject and verb in a sentence. The first rule fetches two words and associated tags from the *nsubj/nsubjpass/csubj/csubjpass* dependencies. If tags in these contain noun and a verb, then we have found the subject and the verb. But sometimes, *nsubj* does not give us the subject and the verb. For example, a sentence *'Bill is an honest man'* , the nsubj dependency gives (bill and man) and the *copula* dependency gives the pair (is and man). Now by rule 5 in the above figure, we can find the subject of the verb by comparing if the dependents of each of the *nsubj* and *cop* are same. Then, we can fetch the subject and verb pair by fetching governors of both dependencies. Further we reject all dependencies which contain anything other than noun type, pronoun type, determiner type and verb type words. Another problem now is, the pronouns and determiners have same tag for singular and plural words. We used the package *inflect* to get plural of a word to see if a PRP or DT is singular or not. Now we have subject verb pairs which needs to be inspected for agreement. Following are the valid tag pairs considered to verify the agreement.

```
valid_sets = [ {'NN','VB'},{'NN','VBD'},{'NN','VBG'},{'NN','VBN'},{'NN','VBZ'},
    {'NNS','VB'},{'NNS','VBD'},{'NNS','VBG'},{'NNS','VBN'},{'NNS','VBP'},
    {'NNP','VB'},{'NNP','VBD'},{'NNP','VBG'},{'NNP','VBN'},{'NNP','VBZ'},
    {'NNPS','VB'},{'NNPS','VBD'},{'NNPS','VBG'},{'NNPS','VBN'},{'NNPS','VBP'},
    {'PRP','VB'},{'PRP','VBD'},{'PRP','VBG'},{'PRP','VBN'},{'PRP','VBZ'},{'PRP','VBP'},
    {'PR$','VB'},{'PR$','VBD'},{'PR$','VBG'},{'PR$','VBN'},{'PR$','VBZ'},{'PR$','VBP'} ]
```

Another paper[3] proposed parsing through the parse tree to extract the subject, verb and object. But the techniques in the paper fail for slightly complicated sentences. Another paper[6] proposed techniques to handle the extraction of subject and verb by handling different sentences like declarative, interrogative in a different way. But the techniques appeared crude and brute force and we decided to skip implementing the idea.

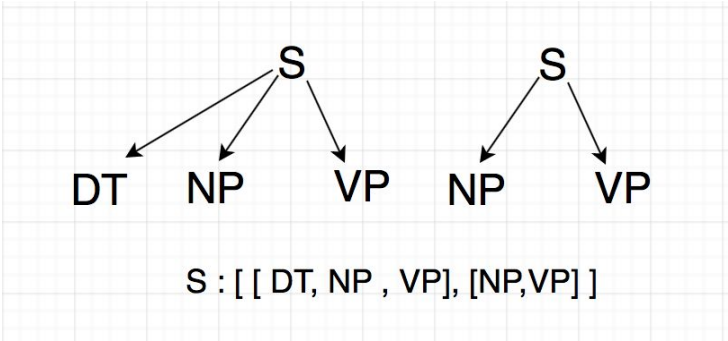
ii) Verb tense / missing verb / extra verb

To figure out missing, extra or incorrect tenses, transitional probabilities are useful as they indicate if a particular verb tag is expected after the previously discovered tags.
For this, we have used an already available corpus *'20newsgroups'*. The categories used for the training were alternative atheism, religion talk, computer graphics and space science. Using this raw data, we calculated the transitional probabilities of every tag against all the tags. This data of probabilities was then stored in a JSON file. Then, each essay is tokenized and tagged which are then read through till we encounter a Verb tag. Once encountered, we look at the precedent and antecedent tag and if any of the tags paired with Verb tag have a probability below the decided threshold, they're considered an error.
We used threshold probability as 0.01; anything below that is considered an error.
Few mistakes are correctly identified, for example, Prob(VBZ | NNS) = 0.01.
But training corpus also had mistakes due to which wrong tags are also accepted, i.e. Prob(VBP | NNP) = 0.06.
Future scope is to find a better corpus with fewer grammatical mistakes.

iii) Sentence Structure

To detect the correct sentence structure, we used the stanford parser to get parse tree for sentences in the essays. Once we have the parse tree, we traverse through the tree and at each node of the tree, we check if the children of the node are valid or not.

To have a reference for valid grammar, we used Vox articles dataset which contains more than 22,000 news articles. We generated parse trees for each sentence in the articles and saved the children for each node in the parse trees. In this way, we have a grammar to look up when we are parsing the trees of essay text. We have the grammar as a dictionary shown below. The dictionary is saved as a pickle file.

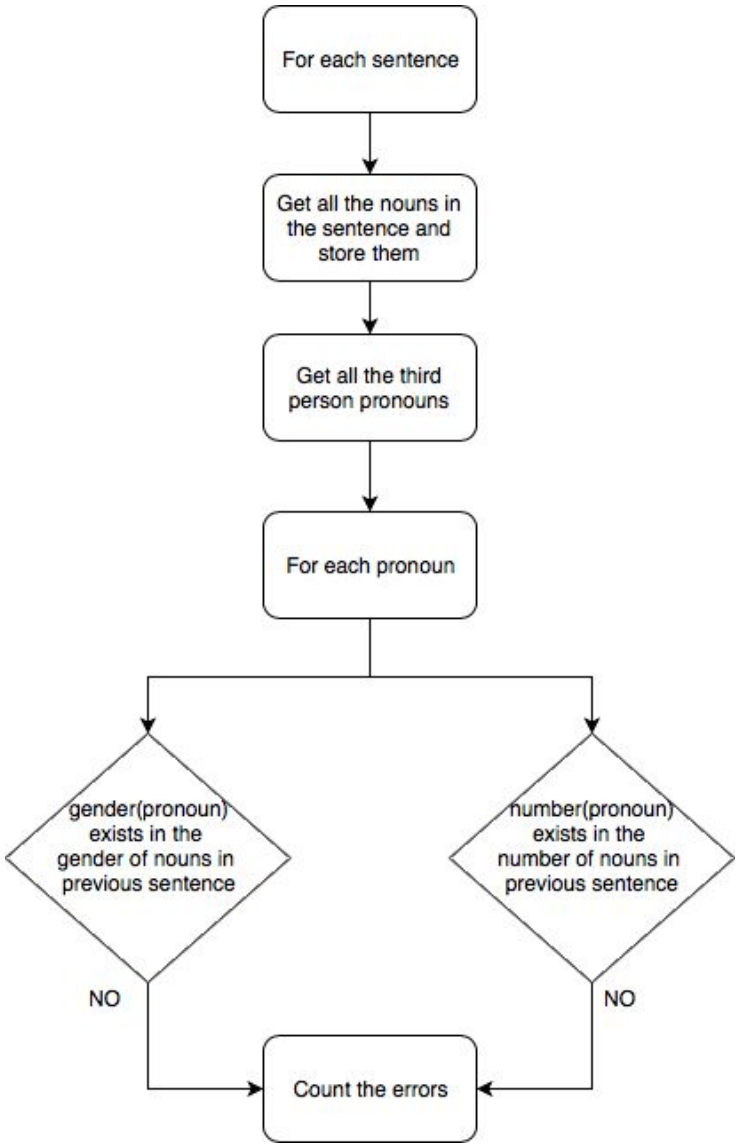



```
{
  S : [ [ NP, VP ], [  DT, NP, VP], [ DT, ADJ, NP, VP] ...],
  NP : [ [ DT, NN], [DT, ADJ, NNS] ...]
  ...
}
```

During testing, we get parse tree for each sentence in the essay and traverse through each node and check if the children of the node are valid from looking up the grammar. While this is not completely accurate as this checks the error in the grammar locally at a node and neither recursively down the tree nor the tree formed by its parents. Grammar check is done only for possibility of a node having certain children.

d) Text Coherence

i) Essay Coherence via pronouns



For each sentence, we get all the nouns and store it for processing in next sentence. For the current sentence, we get all the pronouns. For each pronoun, we check if the gender of the pronoun(male/female/neutral) has a noun in previous sentence matching that gender. If not that's a mistake. Also we check if the number of the pronoun has a noun matching that number. We are using inflect package to find plural of a word to detect the number. If number does not match that's an error. We count the total number of errors in the essay. This number is mapped to the range (1 - 5). The complete flow is represented in the flowchart above.

Recognizing Gender

This part would give us the gender of each token (nouns as well as the pronouns that might be associated with them) as either male, female or neutral. We assessed the gender of each token with respect to its POS tag. There were three types of part of speech possible which were handled in the following ways:

1. **Pronouns**

   We defined a default list of pronouns for male gender such as he, him, his, himself etc and female gender such as she, her, herself. If our target word was tagged pronoun and found in the list of male pronouns, we return gender as male; similarly for female gender. If the word is not found in either of the lists, we return neutral as the gender as those are the only pronouns left such as they, we, them etc.

   ```
   "herself" gender: female
   "s/he" gender: neutral
   ```

2. **Common Nouns**

   We defined a default list of the most common female and male noun words used in new articles. Few female words from the list are 'actress','women','aunt' etc and similarly few common male words are 'grandfather','grandpa','grandson','groom'. Anything except these are checked for their first letter to be capital to rule out the possibility of them being proper noun but wrongly tagged as common noun as the POS tagger didn't recognize the word. If the first letter is in lowercase, we return 'neutral' as the gender. Sample output of the gender module for common nouns:

   ```
   "Fisherman" gender: male
   "elephant" gender: neutral
   ```

3. **Proper Nouns**

   For this, we used NLTK corpus 'names', which gives us a list of female and male names. But there are many names that are not part of NLTK corpus, so we use this data in order train a model for predicting gender based on the last letter of the name/word. We use the entire corpus to train a Naïve Bayesian Classifier which takes the last letter of all the names as feature and name being male or female as the target class. This model is then asked to classify a word which is tagged as Proper Noun but is not present in both, male and female name lists. You can see examples below that gender of non-english names is also being recognized correctly.

   ```
   "Raman" gender: male
   "Sarojini" gender: female
   ```

ii) Topic Coherence

To check the coherence of the essay text with the respective topic, we extract the main words used in the topic by removing all english stopwords (Using NLTK). Then using WordNet, we get all the possible antonyms and synonyms of the main topic words and then count their occurences in the test essay. This count is then normalized by word count of the essay so as to find the relative relevance as if the count of a particular is moderately high wrt to other essays but the essay is extremely lengthier than the others, then the coherence is compromised and should not be considered as high. This normalized value is then scaled to 1 to 5, 1 being the least coherent.

e) Linear combinations

We tried to run a basic Decision Tree learning algorithm on the final score for training data.
With default coefficients provided, we get an accuracy of 65% after performing 10-Fold cross validation.
We tried Linear Regression from sklearn package on the scores from part a, b, c.i and c.ii, c.iii and d.ii to get the coefficients as follows:

Variable	Value	Description
a	0.47805833	Sentence Length
b	-0.1462483	Spelling Errors
c.i	0.01878885	Subject Verb Agreement
c.ii	0.01498653	Verb Usage
c.iii	0.01457708	Sentence Structure
d.i	0.03633159	Text Coherence
d.ii	0.0381478	Topic Coherence

These coefficients gives us an accuracy of 98% with a simple Decision Tree Classifier after performing 10-Fold cross validation.

References:

1. Automated Essay Scoring: Writing Assessment and Instruction
2. Exploring Automated Essay Scoring for Nonnative English Speakers
3. Triplet extraction from sentences
4. Automated assessment of non-native learner essays: Investigating the role of linguistic features
5. IITB System for CoNLL 2013 Shared Task: A Hybrid Approach to Grammatical Error Correction
6. A Light Rule-based Approach to English Subject-Verb Agreement Errors on the Third Person Singular Forms