**ASSIGNMENT 2- CS4248**

**Natural Language Processing**

**Name :** Bathini Yatish **Matric No:** A0091545A

**Introduction:**

The aim of the assignment is to write a program (in Java, C++, or C) to perform context sensitive spelling correction. This task is to detect spelling errors that result in valid words. In this assignment, this task will be formulated as disambiguation between two words in a confusion set. Examples of such confusion sets include { *adapt*, *adopt* }, { *formally*, *formerly* }, { *raise*, *rise* }, etc.

In order to achieve the required context sensitive disambiguation of the words involved, the following features are expected to be used:

a). Surrounding words: Each word that appears in the sentence containing the confusable word *w* is a feature. All surrounding words are converted to lowercase, and stop words and punctuation symbols are removed.

b). Collocations: A collocation *Ci,j* is an ordered sequence of words in the local, narrow context of the confusable word *w*. Offsets *i* and *j* denote the starting and ending positions (relative to *w*) of the sequence, where a negative (positive) offset refers to a word to its left (right). Each collocation string spanning positions *i* to *j* is a feature.

During the course of this assignment, it is expected to arrive at the features that are supposed to be used to achieve the most accurate disambiguation mechanism possible.

**Context Sensitive Spelling Correction:**

Context-sensitive spelling correction is the problem of correcting spelling errors that result in valid words. These errors might arise for several reasons, like typos (e.g. typing adapt instead of adapt) that might occur just due the incorrect entry of a single letter in the word, homonym confusions (their instead of there) or usage errors (peace for piece). Using conventional spell checkers, these errors cannot be detected as the incorrect words are still words that exist in the dictionary.

In order to be able to detect such errors resulting in valid errors, we need to implement a disambiguation checker that chooses the correct disambiguated word based on the context of the sentence. The context of the sentence and the understanding of the usage of the disambiguation words are obtained from the analysis of a training corpus which is a collection of several sentences containing each of the given ambiguous words. The aim is to use the best features possible to try and compute a model statistics files that should be used while later analysing the test corpus and give the disambiguation word that is to be used with each of the sentence with the highest possible accuracy.

**Methods Deployed for Spelling Correction:**

From this section onwards, I shall try to examine the methods that I experimented in order to distinguish between the ambiguous words in the training and the test corpus. The following methods played major roles in the approach that I followed to achieve my goal of disambiguation:

1. **Collocation of words** from a particular index i to another index j relative to the position of the ambiguous word found. i = -2 and j = 2 C*-2,2*
2. **Collocation of words** when the prior collocation of words was not found. *C-2,2* that was earlier used is more accurate but rarely found and hence warranting the use of another form of collocation C*-1,1*
3. **Surrounding context words** that are present within a distance of (+/-) k words. The value of k used in the assignment is 5 and I would like to perform a detailed analysis of how this value of k was determined.
4. **Preceding word** count is used in case all the above methods fail to provide the disambiguated word

Each of these methods involves the same training set and test corpus.

**Collocation of Words(C -2,2):**

A collocation of words Ci,j expresses a set of words that exist between the positions i and j relative to the ambiguous word w. The presence of collocations only with a particular ambiguous word w helps us to understand that the chances of the ambiguous word in a given sentence in the test corpus having the collocation being w is significantly higher than the chances of the ambiguous word not being w.

For the first filter in determining the ambiguous word, the collocation C-2,2 is used. This collocation gives us a collection of 5 words (2 words to the left of the ambiguous word, the ambiguous word and 2 words to the right of the ambiguous word). For each sentence in the training corpus, the counts of each of these collocations are determined for both the ambiguous words and recorded in the model file. It can be safely assumed that in the training corpus, a collocation cannot have an identical count value for both the ambiguous words. This assumption actually makes practical sense since the collocation also defines the context of the sentence which determines the usage of the ambiguous word. In the test corpus, a match when found for 4 words in the collocation(2 words before the ambiguous word and 2 words after the ambiguous word) at the indices -2, -1, 1, 2 are the same as in any collocation that has been stored in the model file, the ambiguous word to be fit into the sentence is merely the central word of this collocation.

Since the training set given to us isn’t very large, the data that is present is insufficient to only rely on this collocation of C-2,2 to determine the disambiguated word and hence further methods are employed to supplement this feature. However, this method was found to be totally consistent with the corresponding answers when matching collocations were found.

**Collocation of Words(C -1,1):**

As explained earlier that the possibility of finding 5 words identical in a training and a test corpus being identical, in order and position with respect to the ambiguous word, is very less and hence another collocation C-1,1 is used for computation.

This collocation gives us a series of 3 words (1 word to the left of the ambiguous word, the ambiguous word and one word to the right of the ambiguous word). From the training corpus it could be observed that certain collocations occurred multiple times with the same ambiguous word. In fact, the number of times that the collocation occurred for a particular ambiguous word was so high that we could be absolutely sure that the words preceding and succeeding the ambiguous word in the collocation always when occurring together also had the same ambiguous word as part of it.

For example, in the {adapt, adopt} training set, the collocation (to adopt a) was found 10 times while (to adapt a) was never found. In such cases we can be absolutely sure that a test corpus having a sentence of the form of (to >> << a) will surely be related to the ambiguous word {adopt}.

However these collocations were again not sufficient to determine the ambiguous words for all the sentences provided but provided a high accuracy rate of {80%, 100%, 100%} for each of the 3 data sets {(adapt,adopt), (formerly, formally), (raise, rise)} whenever a match was found. Hence it is safe to accept the disambiguated words suggested so far in our method.

**Surrounding Words:**

As explained earlier, there is insufficient data given to be able to purely rely on collocations. Hence I came up with another method of being able to disambiguate between the words. For each sentence in the training set, the words surrounding the ambiguous word w is also a feature. These surrounding words when collectively analysed in the test corpus can help us to determine the most suitable word w that goes with it.

In order to calculate the probability of the ambiguous words w1 and w2, a count is kept of each word that is present in the training data and then the Naïve Bayes’ theorem is used to compute these probabilities. Words that are unknown or not found for either of the ambiguous words are ignored in the calculation.





The above formula is used to compute the probability of each ambiguous word given the surrounding words. Here, c is the surrounding word and wi is the ambiguous word. The ambiguous word wi for which the probability calculated is higher, is the ambiguous word that is chosen for that sentence. These surrounding words when used alone helped us achieve a minimum accuracy of 60% for all the data sets

On careful analysis of the training set, it was also noticed that not all the words present in the sentence affected the ambiguous word in the contextual sense. It was noticed that when words went further and further away from the wi, they became less and less relevant to our requirement and could be ignored. Assume a value k, where k is the number of the words to the left and to the right of the ambiguous word that is determined to influence our wi and used to calculate the probability of the ambiguous word, the following data on the accuracy of the results were obtained:

|  |  |  |  |
| --- | --- | --- | --- |
| **Distance = k** | **Adapt/Adopt** | **Formally/Formerly** | **Raise/Rise** |
| K=3 | 60% | 60% | 60% |
| K=4 | 62% | 74% | 74% |
| **K=5** | **70%** | **82%** | **76%** |
| K=6 | 64% | 78% | 68% |
| K=7 | 66% | 80% | 70% |
| K=10 | 62% | 80% | 72% |

Hence from the data obtained, it can be concluded that it is best to use a value of k = (+/-)5 and only keep track of the surrounding words that are a distance of k words from w.

**Stop Words:**

The stop words file that was given for the assignment had a list of words that have a much higher occurrence in the training and the test corpus and most of the words do not syntactically influence the probability of the ambiguous words. However, I tried to perform the tests both before removing and after removing the stop words from the sentences in order to understand the difference in performance caused. The results obtained are listed below:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Adapt/Adopt** | **Formally/Formerly** | **Raise/Rise** |
| Removing stop words | 68% | 60% | 70% |
| **No removing stop words** | **70%** | **82%** | **76%** |

As can be observed from the results obtained, the accuracy of the test/training programs are better before the stop words. Having included stop words and at the same time incorporating the surrounding words feature only for words until a distance of (+/-)k=5 from the ambiguous word, it is ensured that the stop words that are included into the surrounding words as a feature are only those that seem to influence the probability of the ambiguous word

­**Conclusion:**

I adopted a tiered method of using collocations, surround words and count of the preceding word based upon the strength in terms of the accuracy percentages of the different methods. Since C-2,2 has the highest accuracy for the sentences that it is able to predict, it is the first method that is used to decide the ambiguous word. In case such a collocation is not found, then I use the C(-1,1) collocation followed by the probability arrived at using the surrounding words based on the Naive Bayes’ theorem. In case all of the above methods fail, the ambiguous word is chosen based on the preceding word for the index position of the ambiguous word. I tried to use POS tagging as well but was unable to find a pattern by which the POS tagging mechanism could be used to differentiate between the ambiguous words.