CS6370
Natural Language Processing

SPELL CHECK ASSIGNMENT

September 28, 2014

Amal Joy *CS11B006*
Abhijith CS *CS11B003*

1 Introduction

This paper focusses on context sensitive spelling correction.Two methods are popularly used -namely using context words as well as collocations.Context words relies on presence of particular words near the target word.Collocation on the other hand uses POST(part of speech tagging) to determine whether the sequence of word form a valid construct by capturing the local syntax.We did word correction for words,phrases and sentences.For words ,scoring was based on Kernighan and Church's paper on "A Spelling Correction Program Based on a Noisy Channel Model".Whereas for words and phrases,Scoring was provided by "A Bayesian Hybrid method for context-sensitive spelling correction" by Golding.The first and foremost part was indexing the dictionary in not a linear way which could take enormous times.So we used intelligent data Structures like Burkhard-Keller Trees

2 Indexing the Dictinary for fast access

2.1 Burkhard-Keller Trees or BK Trees

It is a tree-based data structure engineered for quickly finding near-matches to a string.They take a problem which has no obvious solution besides brute-force search, and present a simple and elegant method for speeding up searches substantially. In order to index and search our dictionary, we need a way to compare strings. The canonical method for this is the Levenshtein Distance, which takes two strings, and returns a number representing the minimum number of insertions, deletions ,replacements and transpositions required to translate one string into the other. (Since transpositions are taken ,this is actually DamerauLevenshtein distance).Levenshtein Distance forms a Metric Space. Put simply, a metric space is any relationship that adheres to three basic criteria:

1. d(x,y) = 0 $<->$ x = y (If the distance between x and y is 0, then x = y)
2. d(x,y) = d(y,x) (The distance from x to y is the same as the distance from y to x)
3. d(x,y) + d(y,z) >= d(x,z) (Triangle Inequality)

Assume for a moment we have two parameters, query, the string we are using in our search, and n the maximum Levenshtein distance a string can be from query and still be returned. Say we take an arbitary string, test and compare it to query. Call the resultant distance d. Because we know the triangle inequality holds, all our results must have at most distance d+n and at least distance d-n from test.

From here, the construction of a BK-Tree is simple: Each node has a arbitrary number of children, and each edge has a number corresponding to a Levenshtein distance. All the subnodes on the edge numbered n have a Levenshtein distance of exactly n to the parent node. So, for example, if we have a tree with parent node "book" and two child nodes "rook" and "nooks", the edge from "book" to "rook" is numbered 1, and the edge from "book" to "nooks" is numbered 2.

To build the tree from a dictionary, take an arbitrary word and make it the root of your tree. Whenever you want to insert a word, take the Levenshtein distance between your word and the root of the tree, and find the edge with number d(newword,root). Recurse, comparing your query with the child node on that edge, and so on, until there is no child node, at which point you create a new child node and store your new word there. For example, to insert "boon" into the example tree above, we would examine the root, find that d("book", "boon") = 1, and so examine the child on the edge numbered 1, which is the word "rook". We would then calculate the distance d("rook", "boon"), which is 2, and so insert the new word under "rook", with an edge numbered 2.

To query the tree, take the Levenshtein distance from your term to the root, and recursively query every child node numbered between d-n and d+n (inclusive). If the node you are examining is within d of your search term, return it and continue your query.

The tree is N-ary . It appears we can now find misspellings at a O(log n) time, which is better than our O(n) from linear search of dictionary.

2.2 DamerauLevenshtein distance

It is basically the Levenshtein distance but here we also take transpositions into account.It therefore is the minimum number of insertions,deletion,transpositons and substitutions to transform one string into another.We use the concept of dynamic programming to see which is the optimal transformation of a string to another string.

2.3 Preprocessing

Intially we preprocess the browns corpus to get the window of context words for a particular word.We also used Ngrams(basically 5 grams model) to store the context words as well as possible collocations for a word.

2.4 Candidate generation

Here we assumed that a incorrectly spelled word is one which does not exist in dictionary.Therefore when we find a incorrectly spelled word,We report all those word which have a damerau levenschtein distance of 2 from the wrong word and for cases where the word lenght is greater than 7,we provided edit distance threshold of 3 .We basically didnt go for intially 3 edits as we felt it generated a lot of unnecessary candidates and usual spelling mistakes occur at edits of 2.

3 WordSpell

We can often recover a the intended correction c,from a typo t by finding the correction c that maximizes

$$argmax Pr(c)Pr(t|c).$$

Pr(c) is the prior and Pr(t—c) is the likelihood which suggest the probabilty of t being typed by insertions,substitutions,transpositions and deletions when the intended word was c.Together the product is the posterior.We finally choose the candidate which maximises the probability.The dictionary we used intially also had counts of t ,so prior was calculated in preprocessing stage.As for likelihood,we traced back the damerau levenshtein matrix for finding out the which all deletions,insertions,substitutions occured.We assumed two edits are independent.So if we wanted to find Pr(w3|w1) ,we find Pr(w2|w1)Pr(w3|w2) if we are able to transform it to a intermediate words.We used this idea as most of the case there was edits of 2.So for finding "helping" typed as "helpi",the program would convert it to "helpin",So it mean "ng" was typed as "n" and then "in" was typed as "i".The probability of one edit was available online from peter norvig ngrams.In case any of the counts required was available,we used a value of 400 as total occurrences and 20 as count of edit if not available after testing through some trial and error.

By this logic,words with more edits tend to have lesser probability since it results in multiplication of probabilities.This is done for all the candidates and which ever reports the maximum posterior is reported.

3.1 Soundex Test for words

In wordCheck,given a incorrectly typed word,it is hard to ascertain which word was actually especially if the candidates are very similar to incorrectly typed word.So we used the simplest idea of them all ,phonetic intepretation to detect which word must have been intended.We use the mappings of input words to some kind of more general phonetic representation to account for typos that might occur.

Soundex is a well known phonetic algorithm that indexes different words such that homophones get the same encoding.”Robert” and ”Rupert” return the same string ”R163” while ”Rubin” yields ”R150”.But we avoided this later,as we found it was not giving satisfactory results.

Basically all the above listed methods were for Word Spell Check where we didnt have any context information.The next section discusses about phrase Spell check.

3.2 Sample word train data results

gracefull - (graceful=0.009023307, gracefully=0.08774631, grateful=0.0014612668, gratefully=0.0035303035)
ocassion -(occasion=1.1279965E-5, passion=1.9946022E-4, omission=6.943898E-4, occassions=0.0068926434, occassion=0.007191082)
bouyant-(courant=3.9773988E-7, bryant=7.8651495E-4, buoyant=0.074181736)

4 PHRASE SPELL CHECK

For phrases,we started focussing on getting the context into account.We used Brown annotated Corpus with Part of speech tagging as well as Coca Corpus.Word disambiguation was applied for context sensitive spell check.Suppose we have candidates $w1, ...wn$ among which one is the right word and each word w_i is ambiguous with the other in the set.Suppose C=peace,piece ,if spelling program sees occurence of cake,apple etc,it reports piece as context window of piece contains cake etc.Note that the input word must be an incorrect word like pece.If the input is ”a peace of cake”,the program fails to catch it.Context words basically use the idea of particular words within $\pm$k words of the ambiguous target word.If multiple words are correct ,then program first corrects one word and then goes to the next word to get to disambiguate it.The probability of each w_i given the context words c_j observed within a $\pm$ k-word window of the target word.The probability of each w_i is found using bayes rule.

$$p(w_i \mid c_{-k},....c_{-1},c_1,....c_k)=p(c_{-k},...c_{-1},c_1,...c_k \mid w_i)p(w_i)/\ \ p(c_{-k},...c_{-1},c_1,...c_k)$$

However the above likelihood is decomposed into

$$p(c_{-k},...c_{-1},c_1,...c_k \mid w_i)= \prod_{j=-k,...-1,1,..k} p(c_j|w_j)$$

This was found by just counting the number of times c_j occured with w_j divided by total occurences of w_j.However while taking this measure ,we ensured to remove words which occur very common like if ”the” occurs in window of many words,it will not prove as a useful neighbour.Therefore we pruned those words like

```
words.removeAll(Collections.singleton(”the”));
words.removeAll(Collections.singleton(”of”));
words.removeAll(Collections.singleton(”and”));
words.removeAll(Collections.singleton(”a”));
```

And the window size was 3 on both side of the word so as to get sufficient neighbours.However we still faced problems like not getting moon in context window of earth . We found that it was able to attain it at a +5 window from browns corpus.However Coca solved the above problem.

4.1 Phrase scoring

Basically ,we applied the above formula using prior and likelihood.Since the likelihoods are probabilities,and more the number of neighbours,more is the multiplication of probabilites which $\rightarrow 0$ which makes the score smaller.Therefore instead of multipying we summed up the log of probabilities which amounts to maximising the posterior and we also removed prior from the caculation as we felt words which have high prior dominated even if the context neighbour count of these are low.

4.2 sample phrase results

collocation was applied only on coca results as they already gave better results.We didn't do collocation on browns to show what other spurious candidates are generated. The coca results are given priority over brown even if coca scores are lower than browns.
from the eath to the moon
======
coca results areearth=12.0
brown results aredeath=1.0, sat=2.0, teach=1.0

a geant leap for mankind
======
coca tells great with score 42.0 (this indicates that collocation of the word made sense)
coca tells giant with score 75.0
coca results are giant=75.0, great=42.0
brown results are great=2.0, want=1.0

cops and robers
======
coca results are ()
brown results are robbers=1.0
Occasionally browns proved more useful than coca though rare.

rainig cats and doggs
======
coca results are raining=15.0
brown results are
====== coca results are dogs=41.0
brown results are dogs=2.0

We found that for k=3 ,phrase was doing well for browns as beyond 3 ,we were getting lot of unnecessary neighbours which affected the spell check.

5 Sentences

Basically after doing phrases ,sentences were also being corrected by the same logic.However since a sentence is a fully formed construct,we applied part of speech tagging and collocation to resolve ambiguites.Context word are good at capturing nearby neighbours but not on the relative order of them.When order matters,more syntax based methods like collocations and trigrams are appropriate.For example,
roff of the house

The program generated off because off may be in context of house.But "off of the " will not be a valid collocation.A collocation basically expresses a pattern of syntatic elements around the target word.We used words as well as part of speech tag for collocation.

5.1 PART OF SPEECH TAGGING

we went through coca and noted the counts of different sensed for each word.This should be done as a word can have multiple senses like walk can be a noun,verb etc.Then for tagging a sentence,we pick the most general sense of the word.This will not be true all of the time but most of the time it deems to be correct.Once we have a sentence with part of speech tag,we were ready to move to collocations.

5.2 collocation

For a word ,we considered the two adjacent word along with this word .So this word can be at the beginning,middle or the ending.Since part of speech tagging was done ,We also made combinations of the above where the word was substituted by its tag.

For ex w1 w2 w3 wc w4 w5 etc

some of the valid collocations for wc are w2 w3 sc ,w3 wc w4,wc w4 w5 ,wc(t) w4 w5,wc(t) w4(t) w5 etc where wi(t) denoted the most common occuring tag of wi.This was collected from coca and stored in hashmaps for each word for fast access.

5.3 Using collocation for resolving ambiguities

When we have to choose a word in the confusion list,we see that if it provides a valid collocation with its nearest neighbours.If it does it is supported and reported first.for collocation mathcing,we took only tags of nearby neighbours and checked if they are already in the collocation of the above word.However for two or more words in confusion list which have matching collocation ,we havn't resolved the ambiguites as we found it was affecting run time.For this case,the ranking is then based on context word ranking.However if two distinct collocation match for two different words,then we can take the overlap of these two collocations as well number of other matching collocation to rank the words.

5.4 Sample sentence train data results

```
The parliament passed the resoltion to discuss the bil
======
coca results are (resolution=24.0)
brown results are (desolation=1.0, resolution=3.0)
======
coca results are (bill=289.0, hiv=6.0, lie=29.0, sit=22.0)
brown results are (ben=1.0, bow=1.0, hit=1.0, ml=1.0, sit=1.0)

    The fotball match was very interesting
======
coca results are()
brown results are()
In case of no context from coca and browns
possible choice is football with score1.0

    Keep your frinds close and your enemis closer
======
coca tells find with score 125.0
coca results are brings=17.0, find=125.0, friend=523.0, friends=124.0
brown results areblinds=1.0, brings=1.0, find=2.0, friend=6.0, friends=5.0
======
coca results are enemies=7.0, enemy=7.0
brown results areenemy=2.0
```

Since collocation doesnt remove words occuring a lot unlike context word,words like "find" have been reported by collocation as preference whereas pure context reports lesser preference for "find" than friend.

Private hopitals to provide frea treatment to the poor

```
======
coca results are hospital=26.0
brown results are hospital=5.0
======
coca tells few with score 7.0
coca tells great with score 49.0
coca results are area=6.0, feel=10.0, few=7.0, great=49.0, idea=13.0
brown results are feet=1.0, few=1.0, idea=1.0, red=1.0, tree=1.0
```

One thing we noticed was the singular form of the noun was being reported in most of the cases since the context words will be nearly same for both of them .We expected to see "free" treatment but spellchecker reported great treatment which is syntatically valid.

6 HIGHLIGHTS

1. word spell does not still convey the intended word as the background knowledge is not available.So we can use phonetics based algorithms for word spell.
2. context seems to be the most powerful way of correcting words but then we require huge corpora,huge storage,fast look up etc
3. collocation is more useful for valid constructs such as sentences which actually have part of speech tagging.
4. assuming the most common sense of a word is useful in most cases but not always
5. coca corpus seems to be more useful at capturing dependencies than browns.
6. context combined with collocation gives the best results

7 RANKING

The ranking for sentences and phrases were combined finally as we found collocation and context were useful for both the scenarios.If intially coca reports any results ,all the results in that category are given preference as they are results of combined collocation and context.If then coca reports another set of words,there are given next preference as it is based on context.And brown results are given the next preference.Finally if all of them fail to report like the case of "fotball match" in training data,then the one will matching collocations and maximum likelihood from word spell are reported.If all these fail,then word spell results are given for incorrectly typed word.

8 REFERENCES

1. A Spelling Correction Program Based on a Noisy Channel Model - Mark D. Kemighan , Kenneth W. Church, William A. Gale 4
2. Combining Trigram-based and Feature-based Methods for Context-Sensitive Spelling Correction - Andrew R. Golding, Yves Schabes