2018S2 KT Project 1 Peer Reviews

Print Options:

☑ Include Questions & Answers ☑ Include Comments ☑ Include All Reviews ☑ Include File Info

Print

FIRST LAST'S PEERMARK REVIEW OF ANONYMOUS'S PAPER

ASSIGNED QUESTIONS

1. Briefly summarise what the author has done

The author firstly introduced the aim of the report, then briefly presented details of dataset.
The author hypothesis that the typos are because adjacent transpositions within a word and miss-hitting neighboring keys by mistake. Then the author proposed algorithm based on GED that changes the parameter, and also a keyboard effects algorithm that the weight of replacement can be 1 or -1 according to the keyboard, also another approach based on Levenshtein distance introduced a new operation called ”transposition”. Finally, results and analysis for each algorithm are presented, the author concluded that adjacencies transposition typos are commonly made by people because the corresponding algorithm achieved high accuracy, and typical examples are listed.

2. Indicate what you think that the author has done well, and why

The author is very creative to come up with different changes that made to edit distance to
prove his hypothesis, the methodology is explained in great detail, and the changes made to the edit distance is clear and easy to understand. The form of typical examples did very well.
The organization of analysis part of each algorithm is very good.

3. Indicate what you think could have been improved, and why

Firstly, the format of the report is incorrect, Secondly, for the original edit distance part, the author changes parameter but the question mentioned by the author that the algorithm is more likely to get more than one candidates, but it only return the first prediction word is not solved. It actually influence the performance of the algorithm, since one of the prediction result was randomly presented, the accuracy makes no sense. And the algorithm comparison was influenced, the logic is not strict enough.

COMMENTS LIST

No comments added

SUBMITTED FILE INFO

file name	KT_assignment_1.pdf
file size	119.79K

"PROJECT 1" BY ANONYMOUS

COMP90049 Project 1 Report: Waht kinda typoz do poeple mak?

Anonymous

1 Introduction

Spelling errors are not uncommon in everyday life. This report proposes several hypotheses as to how typos are made and tries to find supporting evidence by implementing a prediction system predicated on basic Global Edit Distance algorithm (GED).

There are three datasets being used in this project, namely "wiki_misspell.txt" [1], "wiki_correct.txt" [1], and "dict.txt" [2]. The first two sets refer to a collection of commonly misspelled English words and actually intended corresponding entries respectively. The last dataset servers as the dictionary for approximate string match in the prediction system. However, those datasets have been slightly modified.

Generally, typos occur when people type in excrescent letters, drop a certain letter from intended word, or simply mis-hit neighboring keys by chance. The prediction system is designed by following the concepts as well as theories with respect to approximate sting matching presented in *An Introduction to Information Retrieval* [3].

2 Hypotheses and Methodology

2.1 Hypotheses

Hypothetically, the most common ways of making typos are probably adjacent transpositions within a word. Besides, it is also assumable that people tend to make typos by mis-hitting neighboring keys by mistake.

2.2 Edit Distance

Word prediction programs have been designed based on dynamic-programming global edit distance algorithm. It is slightly different in this system since the parameters for basic operations (m, i, d, r) are (+1,-1,-1,-1) and it calculates the score for transforming a string to another. Thus, a higher score implies higher probability that the corresponding word is truly intended. However, it is more likely to get more than one candidates with same scores. In this case, the system will simply return the first candidate as the prediction word. To test the hypotheses presented above, modifications have been applied to the fundamental edit distance algorithm.

2.3 Methodology

- Keyboard effects — mis-hitting surrounding keys on standard English keyboard. Instead of simply replacing the incorrect letter within a English word and decrease the score by -1, a modified algorithm will first check all the surrounding keys of the misspelled letter on keyboard and then decide whether a penalty will be introduced accordingly. Thus, in our case, the weight of replacement can be either 1 or -1 as it will consider the neighboring incorrect letter being mis-hit by chance. The weights for matching, insertion, deletion and replacement are +1, -1, -1, -1|+1 respectively.

- Transposition of two adjacent letters. This idea derives from Damerau - Levenshtein distance [5]. Apart from the basic operations used in global edit distance, in this case, a new operation called "transposition" has been added into the system. During dynamic-programming calculation, when a single letter mismatch is detected, the system will check the current and its preceding one letter of origin and target strings and decide whether it can be corrected by swapping. If so, the mismatch will not attract any decrease in score as though it is actually a perfect match. The parameters (m, t, i, d, r) are (+1, +1, -1, -1, -1). Regardless of other three operations, parameter "m" represents match and "t" refers to transposition. They weights the same because a perfect match and a swap-to-match are expected equally in our case.

3 Evaluation

3.1 Accuracy

The system aims to output a single result as predicted best match for every single misspelled word. Thus, accuracy is adopted as the only evaluation metric. For each method mentioned above, we will calculate the fraction of correct predictions as its accuracy. Comparing the results with that of the pure edit distance algorithm, an increase in accuracy of prediction is expected so that hypotheses proposed is reasonable to some extent.

3.2 Result

The results are presented as following:

Method	Number of Misspelled	Correct Prediction	Accuracy
Levenshtein distance (0,+1,+1,+1)	4453	2444	54.88%
Modified edit distance (+1,-1,-1,-1)	4453	2901	65.15%
Keyboard Effects (+1,-1,-1,1|-1)	4453	3009	67.57%
Adjacencies transposition (+1,+1,-1,-1,-1)	4453	3174	71.28%

Table 1 Prediction accuracy

3.2.1 Levenshtein distance and Modified edit distance

According to the results above, it is noticeable that Levenshtein distance with (0,+1,+1,+1) gives the poorest prediction, with only 54.88% correct predictions. Whereas the modified global edit distance algorithm with (+1,-1,-1,-1) [5] shows greater performance, with accuracy of 65.15%. It is probably because Levenshtein distance is more focused on the number of operations needed to transform a string to another one, while the modified global edit distance expects matches more. In that case, the modified algorithm is able to correct misspelled words with more matches where Levenshtein algorithm failed. Therefore, further modifications are based on the modified GED algorithm for more accurate outputs. Typical examples are listed in the table below:

Misspelled	Correct	Levenshtein GED	Modified GED
ackward	awkward	☑ awkward	backward
ackward	bawkward	awkward	☑ backward
critized	criticized	critize	☑ criticized
beacuse	because	aeacus	☑ because
dimesnional	dimensional	digestional	☑ dimensional

Table 2 Comparison between Levenshtein GED and Modified GED

3.2.2 Keyboard-related GED

The accuracy of the system has increased to 67.57% from fundamental 65.15%, which is not significant though. The plausible hypothesis that people usually make typos by hitting neighboring wrong keys is probably not feasible with the common misspelled words derived from wikipedia. One possible reason is that the misspelled words listed in the "wiki_misspell.txt" is not primitive enough, which means that mis-hitting is still more likely to be one of the most common types of typos, but the mis-hit characters have already been corrected manually before wikipedia analyze and collect those typos since people tend to be sensitive to misspelled words phonetically.

3.2.3 Adjacencies transposition GED

Referring to table 1, the prediction system saw a relatively significant growth in accuracy by taking adjacent words relationships into account. The proportion of correct prediction words rises from 2901 (by modified GED) to 3174 out of 4453 misspelled entries. Particularly, the correction rate is considerably higher than that of Levenshtein distance algorithm, roughly increased by 30%. Some examples of swap-to-match have been listed out in the table below:

Misspelled	Correct	Modified GED	Adjacencies-transposition GED
almsot	almost	aleshot	☑ almost
archaoelogy	archaeology	archaeogeology	☑ archaeology
baout	about	backout	☑ about
bcak	back	beak	☑ back
becuase	because	bechase	☑ because

Table 3 Comparison between Adjacencies-transposition GED and Modified GED

Actually, this is not an exhaustive table of examples, many other examples can be found in the corresponding output document. The test result suggests that people usually make typos by swapping two adjacent letters.

4 Conclusion

In conclusion, test results indicate that adjacencies transposition typos are commonly made by people. The corresponding GED algorithm achieved an accuracy of 71.28% which is considerably high in practice. However, the datasets used in the test are problematic by and large. For instance, some so-called misspelled

words is actually a real word but not intended such as “hinderance”, which brings down the accuracy somehow. Besides, it is also noticeable that there are some non-word entries being considered as valid and being listed in the dictionary used in this project, which also brings down the accuracy. The algorithm for handling tied candidate corrections is not well-designed as it simply returns the first candidate as the best match.

Reference

[1] En.wikipedia.org. (2018). *Lists of common misspellings*. [online] Available at: https://en.wikipedia.org/w/index.php?title=Wikipedia: Lists_of_common_misspellings&oldid=813410985 [Accessed 27 Aug. 2018].

[2] GitHub. (2018). *dwyl/english-words*. [online] Available at: https://github.com/dwyl/english-words [Accessed 27 Aug. 2018].

[3] Manning, C., Raghavan, P. and Schütze, H. (2008). *An Introduction to Information Retrieval*. New York: Cambridge University Press, pp.56 - 65.

[4] J. Nicholson, "Approximate String Search and Matching", the University of Melbourne, Vic, 2018.

[5] F. Damerau, "A technique for computer detection and correction of spelling errors", *Communications of the ACM*, vol. 7, no. 3, pp. 171-176, 1964.

FIRST LAST'S PEERMARK REVIEW OF ANONYMOUS'S PAPER

ASSIGNED QUESTIONS

1. Briefly summarise what the author has done

   Introduces the algorithm to be used in the report and the datasets that will be used. Mentions a literature but does no summary. Makes a hypothesis and explains the methods used in the report. Introduces two different string matching methods that will be used. Reports the result of four different methods. Lists examples showing the difference in various methods. Concludes the report by comparing the different methods demonstrated in the report.

2. Indicate what you think that the author has done well, and why

   The results are nicely tabulated. It very easy to spot in the report and aesthetically pleasing. The use of examples to demonstrate a point is a nice approach.

3. Indicate what you think could have been improved, and why

   Mentions that the report will follow the design of a previous study but fails to elaborate on what this may be. No need for lots of detail but would be nice to know some idea of what it is. The general sentence structure of the report needs improvement. The flow of the report needs some work. The report mentions the use of two different methods but the reults report four types of methods used. It is also unclear has to how these algorithms determine the prediction of tokens. The examples in the result section are not very clear as to what they are supposed to illustrate. The explanation is not tied to the hypothesis or some other finding. The conclusion is not very clear and most the points should have been discussed previously. The report would have been much clearer and flown better if it had tied its results back to its hypothesis. A stronger and clearer hypothesis should first been established and this should be the centre of analysis and discussion.

COMMENTS LIST

No comments added

SUBMITTED FILE INFO

file name	KT_assignment_1.pdf
file size	119.79K

"PROJECT 1" BY ANONYMOUS

COMP90049 Project 1 Report: Waht kinda typoz do poeple mak?

Anonymous

1 Introduction

Spelling errors are not uncommon in everyday life. This report proposes several hypotheses as to how typos are made and tries to find supporting evidence by implementing a prediction system predicated on basic Global Edit Distance algorithm (GED).

There are three datasets being used in this project, namely "wiki_misspell.txt" [1], "wiki_correct.txt" [1], and "dict.txt" [2]. The first two sets refer to a collection of commonly misspelled English words and actually intended corresponding entries respectively. The last dataset servers as the dictionary for approximate string match in the prediction system. However, those datasets have been slightly modified.

Generally, typos occur when people type in excrescent letters, drop a certain letter from intended word, or simply mis-hit neighboring keys by chance. The prediction system is designed by following the concepts as well as theories with respect to approximate sting matching presented in *An Introduction to Information Retrieval* [3].

2 Hypotheses and Methodology

2.1 Hypotheses

Hypothetically, the most common ways of making typos are probably adjacent transpositions within a word. Besides, it is also assumable that people tend to make typos by mis-hitting neighboring keys by mistake.

2.2 Edit Distance

Word prediction programs have been designed based on dynamic-programming global edit distance algorithm. It is slightly different in this system since the parameters for basic operations (m, i, d, r) are (+1,-1,-1,-1) and it calculates the score for transforming a string to another. Thus, a higher score implies higher probability that the corresponding word is truly intended. However, it is more likely to get more than one candidates with same scores. In this case, the system will simply return the first candidate as the prediction word. To test the hypotheses presented above, modifications have been applied to the fundamental edit distance algorithm.

2.3 Methodology

- Keyboard effects — mis-hitting surrounding keys on standard English keyboard. Instead of simply replacing the incorrect letter within a English word and decrease the score by -1, a modified algorithm will first check all the surrounding keys of the misspelled letter on keyboard and then decide whether a penalty will be introduced accordingly. Thus, in our case, the weight of replacement can be either 1 or -1 as it will consider the neighboring incorrect letter being mis-hit by chance. The weights for matching, insertion, deletion and replacement are +1, -1, -1, -1|+1 respectively.

- Transposition of two adjacent letters. This idea derives from Damerau - Levenshtein distance [5]. Apart from the basic operations used in global edit distance, in this case, a new operation called "transposition" has been added into the system. During dynamic-programming calculation, when a single letter mismatch is detected, the system will check the current and its preceding one letter of origin and target strings and decide whether it can be corrected by swapping. If so, the mismatch will not attract any decrease in score as though it is actually a perfect match. The parameters (m, t, i, d, r) are (+1, +1, -1, -1, -1). Regardless of other three operations, parameter "m" represents match and "t" refers to transposition. They weights the same because a perfect match and a swap-to-match are expected equally in our case.

3 Evaluation

3.1 Accuracy

The system aims to output a single result as predicted best match for every single misspelled word. Thus, accuracy is adopted as the only evaluation metric. For each method mentioned above, we will calculate the fraction of correct predictions as its accuracy. Comparing the results with that of the pure edit distance algorithm, an increase in accuracy of prediction is expected so that hypotheses proposed is reasonable to some extent.

3.2 Result

The results are presented as following:

Method	Number of Misspelled	Correct Prediction	Accuracy
Levenshtein distance (0,+1,+1,+1)	4453	2444	54.88%
Modified edit distance (+1,-1,-1,-1)	4453	2901	65.15%
Keyboard Effects (+1,-1,-1,1|-1)	4453	3009	67.57%
Adjacencies transposition (+1,+1,-1,-1,-1)	4453	3174	71.28%

Table 1 Prediction accuracy

3.2.1 Levenshtein distance and Modified edit distance

According to the results above, it is noticeable that Levenshtein distance with (0,+1,+1,+1) gives the poorest prediction, with only 54.88% correct predictions. Whereas the modified global edit distance algorithm with (+1,-1,-1,-1) [5] shows greater performance, with accuracy of 65.15%. It is probably because Levenshtein distance is more focused on the number of operations needed to transform a string to another one, while the modified global edit distance expects matches more. In that case, the modified algorithm is able to correct misspelled words with more matches where Levenshtein algorithm failed. Therefore, further modifications are based on the modified GED algorithm for more accurate outputs. Typical examples are listed in the table below:

Misspelled	Correct	Levenshtein GED	Modified GED
ackward	awkward	☑ awkward	backward
ackward	bawkward	awkward	☑ backward
critized	criticized	critize	☑ criticized
beacuse	because	aeacus	☑ because
dimesnional	dimensional	digestional	☑ dimensional

Table 2 Comparison between Levenshtein GED and Modified GED

3.2.2 Keyboard-related GED

The accuracy of the system has increased to 67.57% from fundamental 65.15%, which is not significant though. The plausible hypothesis that people usually make typos by hitting neighboring wrong keys is probably not feasible with the common misspelled words derived from wikipedia. One possible reason is that the misspelled words listed in the "wiki_misspell.txt" is not primitive enough, which means that mis-hitting is still more likely to be one of the most common types of typos, but the mis-hit characters have already been corrected manually before wikipedia analyze and collect those typos since people tend to be sensitive to misspelled words phonetically.

3.2.3 Adjacencies transposition GED

Referring to table 1, the prediction system saw a relatively significant growth in accuracy by taking adjacent words relationships into account. The proportion of correct prediction words rises from 2901 (by modified GED) to 3174 out of 4453 misspelled entries. Particularly, the correction rate is considerably higher than that of Levenshtein distance algorithm, roughly increased by 30%. Some examples of swap-to-match have been listed out in the table below:

Misspelled	Correct	Modified GED	Adjacencies-transposition GED
almsot	almost	aleshot	☑ almost
archaoelogy	archaeology	archaeogeology	☑ archaeology
baout	about	backout	☑ about
bcak	back	beak	☑ back
becuase	because	bechase	☑ because

Table 3 Comparison between Adjacencies-transposition GED and Modified GED

Actually, this is not an exhaustive table of examples, many other examples can be found in the corresponding output document. The test result suggests that people usually make typos by swapping two adjacent letters.

4 Conclusion

In conclusion, test results indicate that adjacencies transposition typos are commonly made by people. The corresponding GED algorithm achieved an accuracy of 71.28% which is considerably high in practice. However, the datasets used in the test are problematic by and large. For instance, some so-called misspelled

words is actually a real word but not intended such as “hinderance”, which brings down the accuracy somehow. Besides, it is also noticeable that there are some non-word entries being considered as valid and being listed in the dictionary used in this project, which also brings down the accuracy. The algorithm for handling tied candidate corrections is not well-designed as it simply returns the first candidate as the best match.

Reference

[1] En.wikipedia.org. (2018). *Lists of common misspellings*. [online] Available at: https://en.wikipedia.org/w/index.php?title=Wikipedia: Lists_of_common_misspellings&oldid=813410985 [Accessed 27 Aug. 2018].

[2] GitHub. (2018). *dwyl/english-words*. [online] Available at: https://github.com/dwyl/english-words [Accessed 27 Aug. 2018].

[3] Manning, C., Raghavan, P. and Schütze, H. (2008). *An Introduction to Information Retrieval*. New York: Cambridge University Press, pp.56 - 65.

[4] J. Nicholson, "Approximate String Search and Matching", the University of Melbourne, Vic, 2018.

[5] F. Damerau, "A technique for computer detection and correction of spelling errors", *Communications of the ACM*, vol. 7, no. 3, pp. 171-176, 1964.

FIRST LAST'S PEERMARK REVIEW OF ANONYMOUS'S PAPER

ASSIGNED QUESTIONS

1. Briefly summarise what the author has done
2. Indicate what you think that the author has done well, and why
3. Indicate what you think could have been improved, and why

COMMENTS LIST

No comments added

SUBMITTED FILE INFO

file name	KT_assignment_1.pdf
file size	119.79K

"PROJECT 1" BY ANONYMOUS

COMP90049 Project 1 Report: Waht kinda typoz do poeple mak?

Anonymous

1 Introduction

Spelling errors are not uncommon in everyday life. This report proposes several hypotheses as to how typos are made and tries to find supporting evidence by implementing a prediction system predicated on basic Global Edit Distance algorithm (GED).

There are three datasets being used in this project, namely "wiki_misspell.txt" [1], "wiki_correct.txt" [1], and "dict.txt" [2]. The first two sets refer to a collection of commonly misspelled English words and actually intended corresponding entries respectively. The last dataset servers as the dictionary for approximate string match in the prediction system. However, those datasets have been slightly modified.

Generally, typos occur when people type in excrescent letters, drop a certain letter from intended word, or simply mis-hit neighboring keys by chance. The prediction system is designed by following the concepts as well as theories with respect to approximate sting matching presented in *An Introduction to Information Retrieval* [3].

2 Hypotheses and Methodology

2.1 Hypotheses

Hypothetically, the most common ways of making typos are probably adjacent transpositions within a word. Besides, it is also assumable that people tend to make typos by mis-hitting neighboring keys by mistake.

2.2 Edit Distance

Word prediction programs have been designed based on dynamic-programming global edit distance algorithm. It is slightly different in this system since the parameters for basic operations (m, i, d, r) are (+1,-1,-1,-1) and it calculates the score for transforming a string to another. Thus, a higher score implies higher probability that the corresponding word is truly intended. However, it is more likely to get more than one candidates with same scores. In this case, the system will simply return the first candidate as the prediction word. To test the hypotheses presented above, modifications have been applied to the fundamental edit distance algorithm.

2.3 Methodology

- Keyboard effects — mis-hitting surrounding keys on standard English keyboard. Instead of simply replacing the incorrect letter within a English word and decrease the score by -1, a modified algorithm will first check all the surrounding keys of the misspelled letter on keyboard and then decide whether a penalty will be introduced accordingly. Thus, in our case, the weight of replacement can be either 1 or -1 as it will consider the neighboring incorrect letter being mis-hit by chance. The weights for matching, insertion, deletion and replacement are +1, -1, -1, -1|+1 respectively.

- Transposition of two adjacent letters. This idea derives from Damerau - Levenshtein distance [5]. Apart from the basic operations used in global edit distance, in this case, a new operation called "transposition" has been added into the system. During dynamic-programming calculation, when a single letter mismatch is detected, the system will check the current and its preceding one letter of origin and target strings and decide whether it can be corrected by swapping. If so, the mismatch will not attract any decrease in score as though it is actually a perfect match. The parameters (m, t, i, d, r) are (+1, +1, -1, -1, -1). Regardless of other three operations, parameter "m" represents match and "t" refers to transposition. They weights the same because a perfect match and a swap-to-match are expected equally in our case.

3 Evaluation

3.1 Accuracy

The system aims to output a single result as predicted best match for every single misspelled word. Thus, accuracy is adopted as the only evaluation metric. For each method mentioned above, we will calculate the fraction of correct predictions as its accuracy. Comparing the results with that of the pure edit distance algorithm, an increase in accuracy of prediction is expected so that hypotheses proposed is reasonable to some extent.

3.2 Result

The results are presented as following:

Method	Number of Misspelled	Correct Prediction	Accuracy
Levenshtein distance (0,+1,+1,+1)	4453	2444	54.88%
Modified edit distance (+1,-1,-1,-1)	4453	2901	65.15%
Keyboard Effects (+1,-1,-1,1|-1)	4453	3009	67.57%
Adjacencies transposition (+1,+1,-1,-1,-1)	4453	3174	71.28%

Table 1 Prediction accuracy

3.2.1 Levenshtein distance and Modified edit distance

According to the results above, it is noticeable that Levenshtein distance with (0,+1,+1,+1) gives the poorest prediction, with only 54.88% correct predictions. Whereas the modified global edit distance algorithm with (+1,-1,-1,-1) [5] shows greater performance, with accuracy of 65.15%. It is probably because Levenshtein distance is more focused on the number of operations needed to transform a string to another one, while the modified global edit distance expects matches more. In that case, the modified algorithm is able to correct misspelled words with more matches where Levenshtein algorithm failed. Therefore, further modifications are based on the modified GED algorithm for more accurate outputs. Typical examples are listed in the table below:

Misspelled	Correct	Levenshtein GED	Modified GED
ackward	awkward	☑ awkward	backward
ackward	bawkward	awkward	☑ backward
critized	criticized	critize	☑ criticized
beacuse	because	aeacus	☑ because
dimesnional	dimensional	digestional	☑ dimensional

Table 2 Comparison between Levenshtein GED and Modified GED

3.2.2 Keyboard-related GED

The accuracy of the system has increased to 67.57% from fundamental 65.15%, which is not significant though. The plausible hypothesis that people usually make typos by hitting neighboring wrong keys is probably not feasible with the common misspelled words derived from wikipedia. One possible reason is that the misspelled words listed in the "wiki_misspell.txt" is not primitive enough, which means that mis-hitting is still more likely to be one of the most common types of typos, but the mis-hit characters have already been corrected manually before wikipedia analyze and collect those typos since people tend to be sensitive to misspelled words phonetically.

3.2.3 Adjacencies transposition GED

Referring to table 1, the prediction system saw a relatively significant growth in accuracy by taking adjacent words relationships into account. The proportion of correct prediction words rises from 2901 (by modified GED) to 3174 out of 4453 misspelled entries. Particularly, the correction rate is considerably higher than that of Levenshtein distance algorithm, roughly increased by 30%. Some examples of swap-to-match have been listed out in the table below:

Misspelled	Correct	Modified GED	Adjacencies-transposition GED
almsot	almost	aleshot	☑ almost
archaoelogy	archaeology	archaeogeology	☑ archaeology
baout	about	backout	☑ about
bcak	back	beak	☑ back
becuase	because	bechase	☑ because

Table 3 Comparison between Adjacencies-transposition GED and Modified GED

Actually, this is not an exhaustive table of examples, many other examples can be found in the corresponding output document. The test result suggests that people usually make typos by swapping two adjacent letters.

4 Conclusion

In conclusion, test results indicate that adjacencies transposition typos are commonly made by people. The corresponding GED algorithm achieved an accuracy of 71.28% which is considerably high in practice. However, the datasets used in the test are problematic by and large. For instance, some so-called misspelled

words is actually a real word but not intended such as “hinderance”, which brings down the accuracy somehow. Besides, it is also noticeable that there are some non-word entries being considered as valid and being listed in the dictionary used in this project, which also brings down the accuracy. The algorithm for handling tied candidate corrections is not well-designed as it simply returns the first candidate as the best match.

Reference

[1] En.wikipedia.org. (2018). *Lists of common misspellings*. [online] Available at: https://en.wikipedia.org/w/index.php?title=Wikipedia: Lists_of_common_misspellings&oldid=813410985 [Accessed 27 Aug. 2018].

[2] GitHub. (2018). *dwyl/english-words*. [online] Available at: https://github.com/dwyl/english-words [Accessed 27 Aug. 2018].

[3] Manning, C., Raghavan, P. and Schütze, H. (2008). *An Introduction to Information Retrieval*. New York: Cambridge University Press, pp.56 - 65.

[4] J. Nicholson, "Approximate String Search and Matching", the University of Melbourne, Vic, 2018.

[5] F. Damerau, "A technique for computer detection and correction of spelling errors", *Communications of the ACM*, vol. 7, no. 3, pp. 171-176, 1964.

FIRST LAST'S PEERMARK REVIEW OF ANONYMOUS'S PAPER

ASSIGNED QUESTIONS

1. Briefly summarise what the author has done

   The author proposed two hypotheses about the reason for misspelled words, which are adjacent transpositions and mis-hitting neighbouring keys. wiki_misspell.txt, wiki_correct.txt, and dict.txt are used as data source to verify the hypotheses. The writer used four different algorithms based on edit distance to implement word prediction programs. Because there is only one predicted result of the misspelled word, accuracy is used as the evaluation metric. By analysing the accuracy, the hypothesis which is adjacencies transposition typos are the most common mistake is proved. The author also discussed some possible reasons which may have effects on the accuracy due to the data source we used.

2. Indicate what you think that the author has done well, and why

   The design of the experiments and the results are very good in this report. The author modified the algorithm based on the edit distance correspondingly to verify the hypotheses, which extended the content from the lecture. There is also some evidence showed that the author did some research on the topic like using the Damerau - Levenshtein distance to support the adjacent transpositions idea. Modifying the algorithm according to the hypotheses are very creative, and the analysis in this report is very convincing and reasonable. When the author did the analysis, besides statistics data, the concrete examples are used to make the result analysis easy to understand.

3. Indicate what you think could have been improved, and why

   I think one thing can be improved is that using precision and recall as evolution metrics to evaluate the results. The programs can be modified to return more than one approximate results. Because based of the content in the lecture, precision and recall are more reasonable in most situation.

COMMENTS LIST

No comments added

SUBMITTED FILE INFO

file name	KT_assignment_1.pdf
file size	119.79K

"PROJECT 1" BY ANONYMOUS

COMP90049 Project 1 Report: Waht kinda typoz do poeple mak?

Anonymous

1 Introduction

Spelling errors are not uncommon in everyday life. This report proposes several hypotheses as to how typos are made and tries to find supporting evidence by implementing a prediction system predicated on basic Global Edit Distance algorithm (GED).

There are three datasets being used in this project, namely "wiki_misspell.txt" [1], "wiki_correct.txt" [1], and "dict.txt" [2]. The first two sets refer to a collection of commonly misspelled English words and actually intended corresponding entries respectively. The last dataset servers as the dictionary for approximate string match in the prediction system. However, those datasets have been slightly modified.

Generally, typos occur when people type in excrescent letters, drop a certain letter from intended word, or simply mis-hit neighboring keys by chance. The prediction system is designed by following the concepts as well as theories with respect to approximate sting matching presented in *An Introduction to Information Retrieval* [3].

2 Hypotheses and Methodology

2.1 Hypotheses

Hypothetically, the most common ways of making typos are probably adjacent transpositions within a word. Besides, it is also assumable that people tend to make typos by mis-hitting neighboring keys by mistake.

2.2 Edit Distance

Word prediction programs have been designed based on dynamic-programming global edit distance algorithm. It is slightly different in this system since the parameters for basic operations (m, i, d, r) are (+1,-1,-1,-1) and it calculates the score for transforming a string to another. Thus, a higher score implies higher probability that the corresponding word is truly intended. However, it is more likely to get more than one candidates with same scores. In this case, the system will simply return the first candidate as the prediction word. To test the hypotheses presented above, modifications have been applied to the fundamental edit distance algorithm.

2.3 Methodology

- Keyboard effects — mis-hitting surrounding keys on standard English keyboard. Instead of simply replacing the incorrect letter within a English word and decrease the score by -1, a modified algorithm will first check all the surrounding keys of the misspelled letter on keyboard and then decide whether a penalty will be introduced accordingly. Thus, in our case, the weight of replacement can be either 1 or -1 as it will consider the neighboring incorrect letter being mis-hit by chance. The weights for matching, insertion, deletion and replacement are +1, -1, -1, -1|+1 respectively.

- Transposition of two adjacent letters. This idea derives from Damerau - Levenshtein distance [5]. Apart from the basic operations used in global edit distance, in this case, a new operation called "transposition" has been added into the system. During dynamic-programming calculation, when a single letter mismatch is detected, the system will check the current and its preceding one letter of origin and target strings and decide whether it can be corrected by swapping. If so, the mismatch will not attract any decrease in score as though it is actually a perfect match. The parameters (m, t, i, d, r) are (+1, +1, -1, -1, -1). Regardless of other three operations, parameter "m" represents match and "t" refers to transposition. They weights the same because a perfect match and a swap-to-match are expected equally in our case.

3 Evaluation

3.1 Accuracy

The system aims to output a single result as predicted best match for every single misspelled word. Thus, accuracy is adopted as the only evaluation metric. For each method mentioned above, we will calculate the fraction of correct predictions as its accuracy. Comparing the results with that of the pure edit distance algorithm, an increase in accuracy of prediction is expected so that hypotheses proposed is reasonable to some extent.

3.2 Result

The results are presented as following:

Method	Number of Misspelled	Correct Prediction	Accuracy
Levenshtein distance (0,+1,+1,+1)	4453	2444	54.88%
Modified edit distance (+1,-1,-1,-1)	4453	2901	65.15%
Keyboard Effects (+1,-1,-1,1|-1)	4453	3009	67.57%
Adjacencies transposition (+1,+1,-1,-1,-1)	4453	3174	71.28%

Table 1 Prediction accuracy

3.2.1 Levenshtein distance and Modified edit distance

According to the results above, it is noticeable that Levenshtein distance with (0,+1,+1,+1) gives the poorest prediction, with only 54.88% correct predictions. Whereas the modified global edit distance algorithm with (+1,-1,-1,-1) [5] shows greater performance, with accuracy of 65.15%. It is probably because Levenshtein distance is more focused on the number of operations needed to transform a string to another one, while the modified global edit distance expects matches more. In that case, the modified algorithm is able to correct misspelled words with more matches where Levenshtein algorithm failed. Therefore, further modifications are based on the modified GED algorithm for more accurate outputs. Typical examples are listed in the table below:

Misspelled	Correct	Levenshtein GED	Modified GED
ackward	awkward	☑ awkward	backward
ackward	bawkward	awkward	☑ backward
critized	criticized	critize	☑ criticized
beacuse	because	aeacus	☑ because
dimesnional	dimensional	digestional	☑ dimensional

Table 2 Comparison between Levenshtein GED and Modified GED

3.2.2 Keyboard-related GED

The accuracy of the system has increased to 67.57% from fundamental 65.15%, which is not significant though. The plausible hypothesis that people usually make typos by hitting neighboring wrong keys is probably not feasible with the common misspelled words derived from wikipedia. One possible reason is that the misspelled words listed in the "wiki_misspell.txt" is not primitive enough, which means that mis-hitting is still more likely to be one of the most common types of typos, but the mis-hit characters have already been corrected manually before wikipedia analyze and collect those typos since people tend to be sensitive to misspelled words phonetically.

3.2.3 Adjacencies transposition GED

Referring to table 1, the prediction system saw a relatively significant growth in accuracy by taking adjacent words relationships into account. The proportion of correct prediction words rises from 2901 (by modified GED) to 3174 out of 4453 misspelled entries. Particularly, the correction rate is considerably higher than that of Levenshtein distance algorithm, roughly increased by 30%. Some examples of swap-to-match have been listed out in the table below:

Misspelled	Correct	Modified GED	Adjacencies-transposition GED
almsot	almost	aleshot	☑ almost
archaoelogy	archaeology	archaeogeology	☑ archaeology
baout	about	backout	☑ about
bcak	back	beak	☑ back
becuase	because	bechase	☑ because

Table 3 Comparison between Adjacencies-transposition GED and Modified GED

Actually, this is not an exhaustive table of examples, many other examples can be found in the corresponding output document. The test result suggests that people usually make typos by swapping two adjacent letters.

4 Conclusion

In conclusion, test results indicate that adjacencies transposition typos are commonly made by people. The corresponding GED algorithm achieved an accuracy of 71.28% which is considerably high in practice. However, the datasets used in the test are problematic by and large. For instance, some so-called misspelled

words is actually a real word but not intended such as “hinderance”, which brings down the accuracy somehow. Besides, it is also noticeable that there are some non-word entries being considered as valid and being listed in the dictionary used in this project, which also brings down the accuracy. The algorithm for handling tied candidate corrections is not well-designed as it simply returns the first candidate as the best match.

Reference

[1] En.wikipedia.org. (2018). *Lists of common misspellings*. [online] Available at: https://en.wikipedia.org/w/index.php?title=Wikipedia: Lists_of_common_misspellings&oldid=813410985 [Accessed 27 Aug. 2018].

[2] GitHub. (2018). *dwyl/english-words*. [online] Available at: https://github.com/dwyl/english-words [Accessed 27 Aug. 2018].

[3] Manning, C., Raghavan, P. and Schütze, H. (2008). *An Introduction to Information Retrieval*. New York: Cambridge University Press, pp.56 - 65.

[4] J. Nicholson, "Approximate String Search and Matching", the University of Melbourne, Vic, 2018.

[5] F. Damerau, "A technique for computer detection and correction of spelling errors", *Communications of the ACM*, vol. 7, no. 3, pp. 171-176, 1964.