**Dari Spell Checker**

**Subject: Data Mining**

Submitted by:

Mehreen Najm & Ahmad Zia Yusfi

Department of Information Systems

Computer Science Faculty

Kabul University

Kabul, Afghanistan

**Abstract**

Spell check is a software tool for correcting spelling. It's available in word processing programs, email programs, cell phones, and a variety of other applications, such as blogs and forums. Spell check lets you know when words are misspelled, corrects misspelled words as you type, and allows you to search a whole document for misspelled words.

This project addresses on the problem occurs in modern spell checking: real­ word error detection. This project is based on naïve Bayes algorithm to detect the errors on a paragraph that user enters.

**Contents**

1. **Introduction**
   1. Introduction………..……………………………………………………………………………….…………….1
   2. How does a spell checker works?…..…………………………….…………………………….………1
   3. Algorithm………….………..……………………………………………………………………………………..2
2. **Implementation**
   1. Prepare the dataset…………………………………….……………………………………………………..4
   2. Train model………………………..………………………………………………………….………….………4
   3. Test model………………………………….……………………………………………………………………..4

**4. Conclusion**

4.1 Future Works……………………………………………………………………………………………………...6

4.2 References…………………………………………………………………………………………..…….……….7

**Introduction**

A [software](https://www.computerhope.com/jargon/s/software.htm) program or program feature designed to locate misspelled words and notify the user of the misspellings. Depending on the **spell checker**, the feature may either [autocorrect](https://www.computerhope.com/jargon/a/autocorr.htm) the word or allow the user to select from potential corrections on the misspelled word.

**Spell check** is a software tool for correcting spelling. It's available in word processing programs, email programs, cell phones, and a variety of other applications, such as blogs and forums. Spell check lets you know when words are misspelled, corrects misspelled words as you type, and allows you to search a whole document for misspelled words.

Although spell checkers have become one of the most commonly used features in many programs, they have also become a hindrance. Some people have become so dependent on spell checkers that their spelling and grammar skills have declined. They have a difficult time writing anything correctly without the help of a computer.

**How does a spell checker works?**

The spell checker works by comparing every word typed with a list of thousands of correctly spelled, words and then uses [algorithms](https://www.computerhope.com/jargon/a/algorith.htm) to determine the correct spellings. If a word (e.g., a name), is spelled correctly, you can add it to the program's exceptions list so it will not be flagged as misspelled in the future.

A basic spell checker carries out the following processes:

* It scans the text and extracts the words contained in it.
* It then compares each word with a known list of correctly spelled words (i.e. a dictionary). This might contain just a list of words, or it might also contain additional information, such as hyphenation points or lexical and grammatical attributes.
* An additional step is a language-dependent algorithm for handling [morphology](https://en.wikipedia.org/wiki/Morphology_(linguistics)). Even for a lightly inflected language like [English](https://en.wikipedia.org/wiki/English_language), the spell-checker will need to consider different forms of the same word, such as plurals, verbal forms, [contractions](https://en.wikipedia.org/wiki/Contraction_(grammar)), and [possessives](https://en.wikipedia.org/wiki/Possessive_(linguistics)). For many other languages, such as those featuring agglutination and more complex declension and conjugation, this part of the process is more complicated.

In many classic approach, a spell checker implements a simple dictionary check structure.

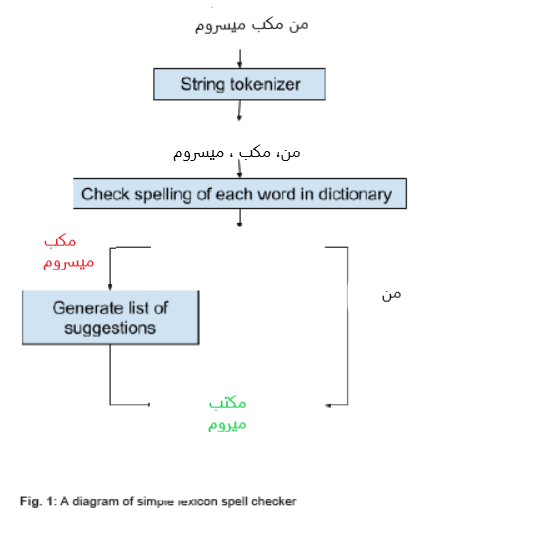
The overall implementation involves simple dictionary check. A diagram demonstrating this

Algorithm is showed in Figure 1.

This method is nice and easy, also requires a low level of programming. Developer can

Simply define a sets of dictionary words for spelling detection and suggestion. If a spelling

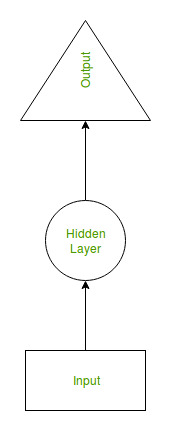
Error occurs, do binary search on the dictionary list and generates a number of corrections. To improve accuracy, simply expanding the dictionary size and it can detect more words.



**Algorithm and Model:**

**Recurrent Neural Networks (RNN)**

**Recurrent Neural Network (RNN)** are a type of [Neural Network](https://www.geeksforgeeks.org/tag/neural-network/) where the **output from previous step are fed as input to the current step**. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus RNN came into existence, which solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is **Hidden state**, which remembers some information about a sequence.

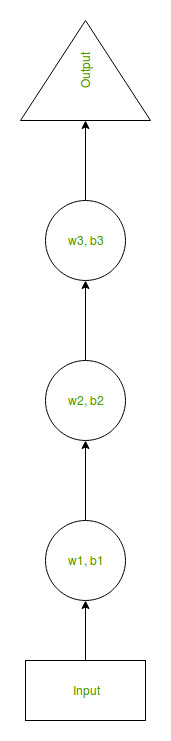


### **How RNN works**

The working of a RNN can be understood with the help of below example:

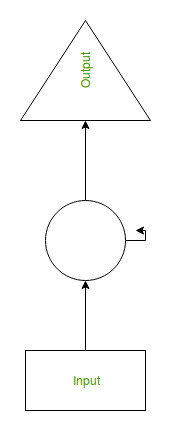
**Example:**

Suppose there is a deeper network with one input layer, three hidden layers and one output layer. Then like other neural networks, each hidden layer will have its own set of weights and biases, let’s say, for hidden layer 1 the weights and biases are (w1, b1), (w2, b2) for second hidden layer and (w3, b3) for third hidden layer. This means that each of these layers are independent of each other, i.e. they do not memorize the previous outputs.



Now the RNN will do the following:

* RNN converts the independent activations into dependent activations by providing the same weights and biases to all the layers, thus reducing the complexity of increasing parameters and memorizing each previous outputs by giving each output as input to the next hidden layer.
* Hence these three layers can be joined together such that the weights and bias of all the hidden layers is the same, into a single recurrent layer.



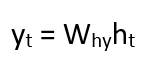
* **Formula for calculating current state:**



1. **Formula for applying Activation function(tanh):**



1. **Formula for calculating output:**



**Implementation**

**Preparing Dataset:**

We have prepared Dari dataset from Darakht-Danish.af online library which they have the most accurate books with correct grammatical syntax. We converted each Dari book into Unicode .txt format.

**Model Implementation:**

The objective of this project is to build a model that can take a Dari sentence with spelling mistakes as input, and output the same sentence, but with the mistakes corrected. The data that we will use for this project will be 32 popular books from Darakht Danish online library. Our model is designed using RNN Algorithm seq2seq Model.

The sections of the project are:

- Loading the Data

- Preparing the Data into correct format that supports Dari

- Building the Model

- Training the Model

- Fixing Custom Sentences

To make things a little more organized, I have put all of the books that we will use in their own folder, called “books”. Here is the function that we will use to load all of the books:

**def load\_book(path):  
 input\_file = os.path.join(path)  
 with open(input\_file) as f:  
 book = f.read()  
 return book**

We will also need the unique file name for each of the books.

**path = './books/'  
book\_files = [f for f in listdir(path) if isfile(join(path, f))]  
book\_files = book\_files[1:]**

When we put these two code blocks together, we will be able to load the text from all of our books into a list.

**books = []  
for book in book\_files:  
 books.append(load\_book(path+book))**

In knowing how many words are in each book, you can use these lines of code:

**for i in range(len(books)):  
 print("There are {} words in {}.".format(len(books[i].split()), book\_files[i]))**

To clean the text of these books is rather simple. Since we will be using characters instead of words as the input to our model, we do not need to worry about removing stop words, or shorten words down to their stems. We only need to remove the English alphabets and characters that we do not want to include and extra spaces.

To track the performance of this model, I will be splitting the data into a training and testing set. The testing set will be composed of 15% of the data.

**training, testing = train\_test\_split(good\_sentences,   
 test\_size = 0.15,   
 random\_state = 2)**

Next step we are going to sort the data by length. This results in sentences of a batch being of similar length, thus less padding is used, and the model will train faster.

**training\_sorted = []  
testing\_sorted = []for i in range(min\_length, max\_length+1):  
 for sentence in training:  
 if len(sentence) == i:  
 training\_sorted.append(sentence)  
 for sentence in testing:  
 if len(sentence) == i:  
 testing\_sorted.append(sentence)**

And here is the full screenshot codes of our project:

**Conclusion:**

This project attempt to solve the spell-checking problem for Dari language. Each suggested method has an advantages and disadvantages and all of them solve the problem. There is may be effected to applied over other languages, and may be merged with each other to implement an optimal spell-checker program for Dari Language.

**References:**

*[1]: (2014) Spelling fixers . Available at: https://schwafire.atavist.com/spellcheck (Accessed: 2*

*May 2016).*

*[2]: Posted and Chitu, A. (2013) Chrome’s contextsensitive*

*spell checker works Offline .*

*Available at:*

*http://googlesystem.blogspot.co.uk/2013/04/chromescontextsensitivespellchecker.*

*html*

*(Accessed: 10 April 2016).*

*[3]: Wikipedia: Lists of common misspellings/A (2015) in Wikipedia . Available at:*

*https://en.wikipedia.org/wiki/Wikipedia:Lists\_of\_common\_misspellings/A (Accessed: 14 April*

*2016).*

*[4]: Damerau, F.J. (1964) A technique for computer detection and correction of spelling errors .*

*[5]: Frith, U. (1980). Unexpected spelling problems*

*[6]: McCloskey, M., Badecker, W., Goodmanschulman,*

*R.A. and Aliminosa, D. (1994) ‘The*

*structure of graphemic representations in spelling: Evidence from a case of acquired dysgraphia’,*

*Cognitive Neuropsychology , 11(3), pp. 341–392. doi: 10.1080/02643299408251979.*

*[7]: Andrew R. Golding (1995), A Bayesian hybrid method for contextsensitive*

*spelling*

*correction*

*[8]: Golding, A.R. and Roth, D. (1999) Machine Learning , 34(1/3), pp. 107–130. doi:*

*10.1023/a:1007545901558.*

*[9]: Jurafsky, D. and Martin, J.H. (2008) Speech and language processing: An introduction to*

*natural language processing, computational linguistics, and speech recognition . 2nd edn. United*

*States: Pearson Prentice Hall.*

*[10]: Drown, D.J., Khoshgoftaar, T.M. and Narayanan, R. (2007) ‘Using evolutionary sampling to*

*mine imbalanced data’, Sixth International Conference on Machine Learning and Applications*

*(ICMLA 2007) , pp. 166 171,*

*. doi: 10.1109/icmla.2007.73.*

*[11]: Bassil, Y. and Alwani, M. (2012) ‘Contextsensitive*

*spelling correction using Google web 1T*

*5Gram*

*information’, Computer and Information Science , 5(3). doi: 10.5539/cis.v5n3p37.*

*[12]: C. Whitelaw, B. Hutchinson, GY Chung, G. Ellis (2009), Using the Web for Language*

*Independent Spellchecking and Autocorrection*