# Spell-Checking based on Syllabification and Character-level Graphs for a Peruvian Agglutinative Language

**Carlo Alva**
Facultad de Ciencias e Ingeniería
Pontificia Universidad Católica del Perú
carlo.alva@pucp.pe

**Arturo Oncevay-Marcos**
Departamento de Ingeniería, GRPIAA
Pontificia Universidad Católica del Perú
arturo.oncevay@pucp.edu.pe

## Abstract

There are several native languages in Peru which are mostly agglutinative. These languages are transmitted from generation to generation mainly in oral form, causing different forms of writing across different communities. For this reason, there are recent efforts to standardize the spelling in the written texts, and it would be beneficial to support these tasks with an automatic tool such as a spell-checker. In this way, this spelling corrector is being developed based on two steps: an automatic rule-based syllabification method and a character-level graph to detect the degree of error in a misspelled word. The experiments were realized on Shipibo-konibo, a highly agglutinative and Amazonian language, and the results obtained have been promising in a dataset built for the purpose.

## 1 Introduction

In Peru, there are several native languages through the diverse native communities in the Amazonian region, such as Asháninka, Kakataibo, Shipibo-konibo, among others (Rivera, 2001). These languages, in spite of being very different from each other (47 languages in 21 linguistic families), share some features related to their morphology and the context in which they are used.

Regarding the morphology of the Amazonian languages, they are highly agglutinative, where suffixes predominates over prefixes. This characteristic distances them a lot from Spanish, the main official language in the country, and even the structural order is also different.

On the other side, these languages are used and transmitted mainly in an oral way, such as storytelling, poetry and in everyday life in the native communities. This causes differences in the way of writing between communities, and even among people in the same community (Aikman, 1999). For this reason, the texts that were written in these languages did not have an orthographic standard to guide them.

Thus, it is a must to support the educational process of this languages for this communities, and from the computational side, the main way to help them would be through the development of automatic tools or functions that process the specific language, in order to assist tasks related to generate written material such as educational books.

In that way, this project aims to develop a spell-checker focused on Shipibo-konibo, an amazonian language that is one of the most studied by linguists (Valenzuela, 2003) and also there are efforts from the computer science field to develop a basic toolkit for it (Pereira et al., 2017).

As this kind of language possess a rich morphology, the spelling corrector would focus on process sub words parts, such as syllables and characters, developing data structures and functions that could help in the process of identifying a misspelled word and suggesting potential corrected alternatives.

This study is organize as follows: In the next section, there will be described some studies related to the implementation of spelling correctors focusing on low-resource languages or language-independent models. Then, the sub word approach for the resources used (data structures and functions) will be detailed. After that, Section 4 describes the proposed spelling corrector, while Section 5 presents the experimentation and results obtained. Finally, the conclusions and future work are discussed in Section 6.



*Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 109–116,
Copenhagen, Denmark, September 7, 2017. 

## 2 Related work

The related works focus on studies regarding the development of spell-checkers in a low-resource scenario or with a language independent approach.

Firstly, Barari and QasemiZadeh presented a tool called "CloniZER Spell Checker Adaptive". It consists of an adaptive method that uses internal error pattern recognition based on a ternary search tree. Thanks to this approach, the spell-checker was independent of the language because it was not based on specific rules from a specific language or corpus (this could be replaced). An interesting part in this approach resides is the support of the tree with variable weighted edges. The weights modifications are made through the interaction with a user, since the method learns from a mean of error patterns and thus the suggestion of solutions keeps improving.

Another source found is Abdullah and Rahman, who performed a generic spelling correction engine for South Asian languages, which uses circular lists where words are grouped by phonetic similarity and with an algorithm adapted from the Recursive Simulation (Lee, 1997) is constructed the possibly correct word that has similar to the misspelled word. However an interesting help they used was an additional dictionary of words, where they kept the misspelled words that the users wrote. This method is favorable for languages that have similarity with other phonetically, such as the group of languages of the Pano family in which the shipibo-konibo is.

Aduriz et al., who presented a corrector based on morphologies, in which a morphological analysis is used to perform morphological decompositions at two levels for spelling errors and for typographical errors uses a recognition of morphemes in the generation of correct words. This approach is interesting since they additionally use a lexicon that they are improving and a set of rules that help to map the lexical level and the surface level due to the morphological transformation (phonological representation of the morphemes).

Finally, Wasala et al. presented a proofreader for an African language. In this it was used a statistical model based on n-grams, this approach is based on assigning probabilities to a sequence of words where the sequence is determined by n-gram. An example of a 2-gram or bigram is: "try this". The chosen approach offers relative ease of construction and avoids the problem of having few linguistic resources. The algorithm that is proposed for the orthographic correction uses 4 modules. These are: pre-processing, generation of permutations, selection of best suggestions, and post-processing. The interesting thing about this algorithm is that after preprocessing it performs a word search with similar sounds or phonemes, thus generating possible solutions, which are then improved based on the statistics of n-grams applying from unigramas to trigrams.

| Vowels |
|---|
| a |
| e |
| i |
| o |

| Consonants |
|---|
| b |
| k |
| ch |
| j |
| m |
| n |
| p |
| r |
| s |
| sh |
| x |
| ts |
| w |
| y |

Table 1: Vowels and consonants in shipibo-konibo.

## 3 Subword Resources

As a first step, it is necessary to specify what type of resources, such as data structures, were used for the development of the spell checker.

### 3.1 Character-level Graph

The first structure that is needed is a graph as is shown in Figure 1. The nodes represent the characters that are used in the Shipibo-konibo (SHP) vocabulary, while the vertexes are the (weighted) relationships between each pair of them (this information is extracted from a corpus). Specifically, the alphabet of SHP contains 4 vowels and 15 consonants, as it can be seen in Table1. It is important to note that all the nodes will not be used in the whole process, due to the proposed n-gram based approach.

### 3.2 Syllable-level Graph

Another structure, needed to improve the possible algorithm solution, is a syllable-level graph. The nodes are the syllables that could be formed in the SHP grammar, and the vertexes represent the potential proximity relationship between 2 syllables (that is extracted from a corpus also). The number

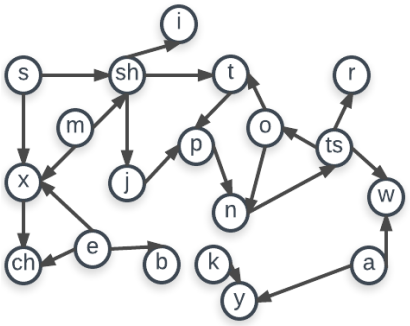


Figure 1: Character-level Graph Structure

| Attack | Core | Coda |
|---|---|---|
| | Vowel | |
| Consonant | Vowel | |
| | Vowel | Consonant |
| Consonant | Vowel | Consonant |

Table 2: Syllabic pattern

of grammatically correct syllables in SHP is 576, and with these syllables, all the word entries of a SHP dictionary could be generated.

#### 3.2.1 Syllabification function

There are 4 syllabic patterns in the SHP language, and these are represented in Table 2. There are 3 positions named Attack, Core and Coda.

The syllabification function helps in the improvement of the solutions selection of the correction algorithm. It has been developed based on the existing rules in the Shipibo-konibo grammar, and it helped the process because it allowed to separate each word of the dictionary in syllables to create the syllables graph. In addition, the use of the syllabification functions is a filter that is used to identify whether a word is well written or not.

### 3.3 Dictionary for previous corrections

As another needed resource, there is an own built dictionary structure that saves the previous misspelled word that has already been corrected, in order to avoid the same error correction again. The key is the misspelled word, and this is associated with a list of words corrected previously as is shown in Figure 2.

## 4 Proposed Spelling Corrector

First, the text is tokenized. After that, there is a verification process for each word to know if the term have been corrected before, or belongs to the

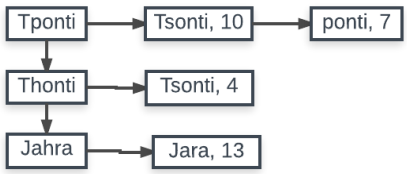


Figure 2: Sample of a misspelled-corrected dictionary

Spanish language, or exceeds a modified language model. In those cases the suggestion is sent directly as it can be appreciated in Figure 3. In other case, the word goes to the correction process, in which a graph is traversed in order to be able to identify possible correct words. Also, a filter is used where the word must be syllabified and finally a score is applied while traversing a graph of syllables and other score with the edit-distance. The results are ranked by score and assigned as suggestions to each corrected word.

### 4.1 Spell-checking algorithm

#### 4.1.1 Identifying a misspelled word

First, the text is tokenized, and the numbers and punctuation are removed. The position of these filtered characters are saved, in order to replace them after the correction process is complete. Besides, the text is transformed to lowercase letters and the accent marks are removed.

As there is a dictionary structure that stores previous corrections, a search of the misspelled word is performed.

If the terms are not found in the previous dictionary structure, it is necessary to identify if they belong to the Spanish language using a corpus(Davies, 2002), a feature shared with other native languages in Perú.

Words that are not found in this Spanish corpus, are evaluated by the syllable language model. This model sum-up the weights assigned by each syllable found in the word, if this value can't surpass an empirical calculated threshold, is marked as a misspelled word an it will be the input for the next stage.

#### 4.1.2 Correcting a misspelled word

For this stage, both graphs of letter and syllables described in the previous section are loaded. Each word that has to be corrected is evaluated. In this way, the number of vowels in the word allows to approximate number of syllables that can be iden-

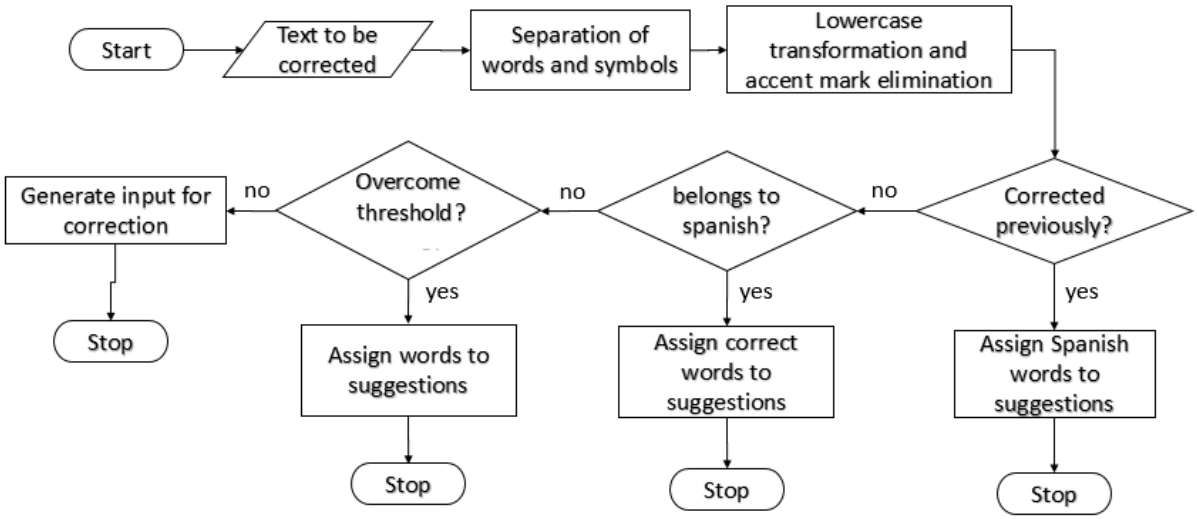


Figure 3: Identifying a misspelled word

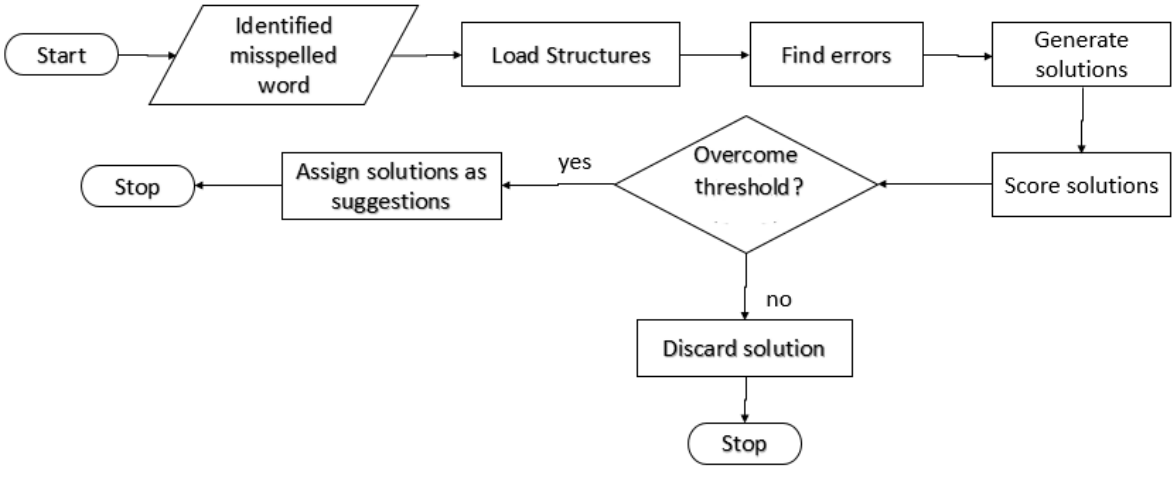


Figure 4: Correcting a misspelled word

tified. Since the absence or increasing of a vowel may be an error, it is considered a little higher range of syllables that can form the word: [number of syllables - 1; number of syllables + 1]. This helps in the generation of the possible correct solutions of each word.

The next step is the solution search. This is done by recursively traversing the misspelled word letter by letter. In this way, each possible syllable combination is contrasted versus the rule-based syllabification model and the range of syllables considered. Also, each solution receive a value calculated by the sum of the repetition of each own letter in the dictionary and the value of the unions in the graph. While traversing the misspelled word by finding a mistake, different paths are generated, the first one is from deleting the wrong letter, the next is when making a change of letter with the previous one, and the other paths are generated by changing the wrong letter with related letters in the graph. In this way, when creating several paths are generating different solutions.

Once the search for correct solutions is completed, they are evaluated through a modified language model. For that purpose the second graph containing syllables and the repetitions of the connections between syllables are used. In order to start the process, each possible solution is separated into syllables and the sum of the connections of these syllables is calculated using the graph. Additionally, the edit distance with Damerau-Levenshtein is calculated. At the end of calculating the two values, a 90% multiplication was performed on Damerau-Levenshtein and 10% on the sum of syllabic relationships, these values of 90% and 10% were chosen after testing to identify which optimized results. Finalized the calculations to each word, there are chosen the three possible correct words with the best values to be returned as solution.

### 4.2 Suggestions component

The suggestions component has been defined so that the user can make corrections to the solution that the application provides, allowing to improve the results since it will not have a totally accurate correction and adding new elements to the corrected list structure. It starts when the user selects the corrected word, activating a menu with the additional suggested words that were obtained when performing the correction. Then, the user selects the option that seems more precise and it is changed in the corrected text. Once this change is made, an additional change is made to the internal structures, where the correct word is added to the corrected list structure and the values are updated in the internal graph that is handled for the correction algorithm.

## 5 Experimentation and Results

An experiment will be carried out to establish the effectiveness of the spelling checker using metrics (van Huyssteen et al., 2004) to evaluate this type of projects.

### 5.1 Dataset

To construct the dataset there was more than one source. The first is a dictionary of shipibo-konibo and Spanish which through a preprocessing has been updated to the new rules of writing and consists of 5486 words. The second source is separated texts by domain (educational, legal and religious) that was translated from Spanish to Shipibo-Konibo. The educational domain consists of 2097 sentences consisting of 16789 words, the legal domain consists of 957 sentences consisting of 16794 words and the religious domain consists of 13482 sentences consisting of 212921 words. This is the initial dataset that helps to construct the graphs that are needed in the correction algorithm.

The dataset is available at a website project.[1]

### 5.2 Design of the experiment

To perform the experiment, where the effectiveness of the algorithm will be tested, it has been decided to use different sentences extracted from the shipibo-konibo dictionary (Lauriout et al., 1993). These sentences correspond to the examples of each dictionary entry. As the dictionary is not with the new official changes, proceeded to perform a pre-processing to update all the words in the example sentences.

After cleaning, 2 types of tests will be generated. The first test is to randomly add a character to some words in the sentence. On the other hand, the second test adds, deletes or changes characters of some words at random in the sentence.

Three columns are created in a table: in the first column the original sentence which is already cleaned, in the second column the sentence with the first type of error and in the last column the

[1] chana.inf.pucp.edu.pe/resources

| Sentence | Sentence for test type 1 |
|---|---|
| nokon wái óroa pekaora ea náshiai | nyokon wái aóroa poekaora eda náshiai |
| eara nénoa iki | eara nénopa iki |
| rámara títa ka cónko iki | rámara títaa ka cónko ikii |

Table 3: Example of sentences for the test 1

| Sentence | Sentence for test type 2 |
|---|---|
| nokon wái óroa pekaora ea náshiai | nokon wáji óra dpekaora a náshiai |
| eara nénoa iki | eaora néoa oiki |
| rámara títa ka cónko iki | rámara títa k cónko iki |

Table 4: Example of sentences for the test 2

sentence with more than 1 type of error. The file with the generated tables is used as input to perform the experiment.

### 5.3 Results

Correction of the 2 types of test was done with 5121 sentences. In order to identify the correct functioning, the Recall, Precision, measure of suggestions and general performance metrics were proposed for the evaluation of spell checkers (van Huyssteen et al., 2004). When counting the number of words in sentences, the result was 55786 words, these included correct words and misspelled words which will allow a better evaluation of the metrics. To calculate the recall and precision of the results:

- True positives (Tp): Misspelled word that are well corrected.
- True negatives (Tn): Misspelled word that are poorly corrected.
- False positives (Fp): Correct word that are well corrected.
- False negatives (Fn): Correct word that are poorly corrected.

To calculate the measure of suggestions a score is used depending on the suggestion of correction that will be applied to all corrections. Upon completion, a sum of all the scores obtained from the corrections will be made and divided by the number of positive ones to find the value of the suggestion measure. The scores used are:

- Correction in the first position of the suggestions: 1 point

| Data | Value |
|---|---|
| True positive | 7099 |
| True negative | 6276 |
| False positive | 14200 |
| False negative | 128 |
| Correction in the first position of the suggestion | 13340 |
| Correction in some position of the suggestions | 7959 |
| No correct suggestion | 6404 |
| No suggestion | 533 |

Table 5: Experiment type 1 data

| Data | Value |
|---|---|
| True positive | 3504 |
| True negative | 9769 |
| False positive | 14242 |
| False negative | 111 |
| Correction in the first position of the suggestion | 12695 |
| Correction in some position of the suggestions | 5051 |
| No correct suggestion | 9980 |
| No suggestion | 510 |

Table 6: Experiment type 2 data

- Correction in some position of the suggestions: 0.5 points
- No correct suggestion: -0.5 points
- No suggestion: 0 points

With the values in Tables 5 and 6, it was possible to calculate the recall and precision metrics with the formulas 1 and 2:

$$recall = \frac{Tp}{Tp + Fp} \tag{1}$$

$$precision = \frac{Tp}{Tp + Tn} \tag{2}$$

Finally, to find the value of overall performance of the corrector, the following formula 3 is used:

$$overall = \frac{Tp + Fp}{Tp + Tn + Fp + Fn} \tag{3}$$

,

With the results obtained in Table 7 and Table 8, the values of the Recall and precision metrics are low, however this is because the spell-checker has yet to be improved to better identify the errors.

| Data | Value |
|---|---|
| Recall | 0.33 |
| Precision | 0.53 |
| Suggested Measure | 14117.5 points |
| Overall Performance | 0.76 |

Table 7: Resulting metric type 1

| Data | Value |
|---|---|
| Recall | 0.19 |
| Precision | 0.26 |
| Suggested Measure | 10230.5 points |
| Overall Performance | 0.64 |

Table 8: Resulting metric type 2

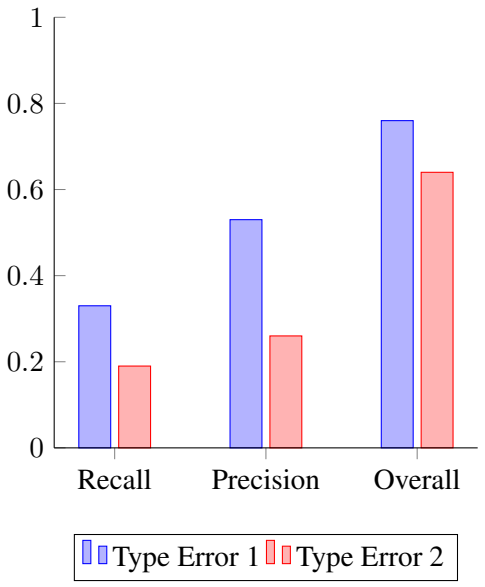


Figure 5: Metrics by type of error

What would improve its effectiveness is to be able to better detect words that do not need to be really corrected, because as can be appreciate, many of the words that are corrected are words that did not need it. An approach to face this problem is to take advantage of the available corpus to perform a search of the word and identify in the corpus if it already exists. This would avoid unnecessary correction, but would mean an increase in the time due to a search for each word that although in short texts would not be much difference, in longer texts it would be noticed.

Despite the low numbers in recall and precision as can see in Figure 5, the spell-checker get a good result at the general level because it is considered within the correct result that both misspelled and well written words, when corrected, offer a correct result. In addition, it can be seen that more than half the time a good correction proposal has been found and 25% of the time these corrections are in the first position as a suggestion in the two types of tests performed.

There was another experiment where it can be found the ranking of the suggestions, for this case

| Top | Type of errors 1 | Type of errors 2 |
|---|---|---|
| 1 | 13400 | 12695 |
| 3 | 722 | 376 |
| 5 | 665 | 337 |
| 7 | 6572 | 4338 |

Table 9: Number of words in top suggestions by type of errors

we use four elements: the first position, the top-3, the top-5 and the top-7. These results can be seen in Table 9. In most cases, the corrected suggestions are in the first positions and low values in the next positions in both types of errors.

## 6 Conclusions and Future Work

In this study, it was proposed a hybrid approach for the development of a spell-checker for Shipibo-konibo. This method was supported with the implementation of linguistic rules (in the syllabification process) and the information obtained from different text corpus for the language. One of the difficult tasks was the use of recursion at the following of different paths when finding an error in the words (since it could be possible to change the character, add or remove it). That is why it was used a cutoff depth that help not to create long paths.

Finally the results were obtained, however they were not very promising because the approach followed is not enough to obtain a precise correction. However, as a future work, a context analysis will be integrated, using embedded words with a character-level model to identify terms that should not be corrected, because they may be well written but are not suitable in the context of a sentence.

### Acknowledgments

For this study, the authors acknowledge the support of the “Consejo Nacional de Ciencia, Tecnología e Innovación Tecnológica” (CONCYTEC Perú) under the contract 225-2015-FONDECYT.

## References

ABA Abdullah and Ashfaq Rahman. 2003. A generic spell checker engine for south asian languages. In *Conference on Software Engineering and Applications (SEA 2003)*. pages 3–5.

Itziar Aduriz, MIRIAM Urkia, INAKI Alegria, XABIER Artola, NEREA Ezeiza, and KEPA Sarasola. 1997. A spelling corrector for basque based on

morphology. *Literary and Linguistic Computing* 12(1):31–38.

Sheila Aikman. 1999. *Intercultural education and literacy: An ethnographic study of indigenous knowledge and learning in the Peruvian Amazon*, volume 7. John Benjamins Publishing.

Loghman Barari and Behrang QasemiZadeh. 2005. CloniZER spell checker adaptive language independent spell checker. In *AIML 2005 Conference CICC, Cairo, Egypt*. pages 19–21.

M Davies. 2002. Corpus del español: 100 million words, 1200s-1900s. *Available online at* `http://www.corpusdelespanol.org` .

Erwin Lauriout, Dwight Day, and James Loriot. 1993. Diccionario shipibo-castellano .

Lung-fei Lee. 1997. A smooth likelihood simulator for dynamic disequilibrium models. *Journal of econometrics* 78(2):257–294.

Jose Pereira, Rodolfo Mercado, Andres Melgar, Marco Sobrevilla-Cabezudo, and Arturo Oncevay-Marcos. 2017. Ship-lemmatagger: building an NLP toolkit for a peruvian native language. In *Text, Speech, and Dialogue: 20th International Conference, TSD 2017*. Springer (In-press).

Andrés Chirinos Rivera. 2001. *Atlas lingüístico del Perú*, volume 6. Centro Bartolomé de Las Casas.

Pilar Valenzuela. 2003. *Transitivity in shipibo-konibo grammar*. Ph.D. thesis, University of Oregon.

Gerhard B van Huyssteen, ER Eiselen, and MJ Puttkammer. 2004. Re-evaluating evaluation metrics for spelling checker evaluations. In *Proceedings of First Workshop on International Proofing Tools and Language Technologies*. pages 91–99.

Ruwan Asanka Wasala, Ruwan Weerasinghe, Randil Pushpananda, Chamila Liyanage, and Eranga Jayalatharachchi. 2011. An open-source data driven spell checker for sinhala. *ICTer* 3(1).