Normalization of Numeronyms using NLP Techniques

Avishek Garain
Computer Science and Engineering
Jadavpur University
Kolkata, India
avishekgarain@gmail.com

Sainik Kumar Mahata
Computer Science and Engineering
Jadavpur University
Kolkata, India
sainik.mahata@gmail.com

Subhabrata Dutta
Computer Science and Engineering
Jadavpur University
Kolkata, India
subha0009@gmail.com

***Abstract*—This paper presents a method to apply Natural Language Processing for normalizing numeronyms to make them understandable by humans. We try to deal with the problem using two approaches, viz., semi-supervised approach and supervised approach. For the semi-supervised approach, we make use of the state of the art DamerauLevenshtein distance of words. We then apply Cosine Similarity for selection of the normalized text and reach greater accuracy in solving the problem. For the supervised approach, we used a deep learning architecture to solve the problem at hand. Our approach garners accuracy figures of 71% and 72% for Bengali and English (for the semi-supervised approach) and 89% for the supervised approach, respectively.**

***Index Terms*—Cosine Similarity, Damerau Levenshtein Distance, LSTM, Numeronym**

I. Introduction

A numeronym is a number-based word. Most commonly, a numeronym is a word where a number is used to form an abbreviation [1], [2], [7]. Pronouncing the letters and numbers may sound similar to the full word: ”K9” for ”canine” (phonetically: ”kay” + ”nine”).

Nowadays, the use of numeronyms is widespread due to the concept of Language Localization. Language localization is the process of adapting the product in the language suited to the particular culture and geographical location/market. The need to communicate and connect with the younger audience is the main reason to adopt language localization services.

There is a thin line between localization and translation. Translation includes grammar and spelling issues which vary by the geographical locations. Localization deals more with significant, non-textural components of products or services. It addresses other aspects such as adapting graphics, using appropriate date and time formats, adopting the local currency, choices of colors, and cultural references amongst many other details.

Now, when short segments of alphabets are replaced by numbers, the resulting word is still readable, but more often, we can see a more complex form of numeronyms, such as L10N(Localization) and I18N(Internationalization). Deciphering these form can be quite trivial and needs an acquaintance with such language. To the best of our knowledge, we did not find any previous state-of-art work done in this domain.

We have used two approaches to solve the problem. The first approach is a semi-supervised method, where, to find the normalized version of the words, we have used the concept of Damerau-Levenshtein [3] distance and Cosine similarity. This approach can counter the problem to a large instance and when checked manually, it gives us an accuracy of 71% and 72%, for Bengali and English respectively.

The second approach is based on Deep Learning architecture, where numeronyms and their corresponding normalized words are given to a neural network in character embedding. The network then learns to generate normalized words of a given numeronym. This approach, when checked manually, gives us an accuracy of 89% for both the languages.

The rest of the paper is organized as follows. Section II defines the data that was used for the experiment. The working of the algorithm has been described in detail in Section III. This is followed by results and concluding remarks in Section IV and V respectively.

II. Datasets

As such no state of the art corpus consisting of numeronym words and their corresponding normalized words are available. So we developed our data-set by collecting 8000 English and 2000 Bengali numeronym words from various digital resources. These data-sets were termed as Nm_e (English) and Nm_b (Bengali). Also, a large number of English and Bengali Wikipedia pages were scrapped and the data from them were tokenized, using Stanford Tokenizer[1]. This data was used as the generative data for the semi-supervised approach. Let us call this data as D_e (English) and D_b (Bengali). Also, the collected numeronyms were converted to its normalized form using human effort. This manually normalized data would be used as our generative data for the supervised approach.

III. Methodology

To cater to the problem at hand, we used two methods, a semi-supervised approach and supervised approach.

[1]https://nlp.stanford.edu/software/tokenizer.html

ISBN: 978-1-7281-4283-8 PART NO.: CFP20O01-ART

A. Semi-Supervised Approach

For the semi-supervised approach, initially the collected English and Bengali sentences from Wikipedia, were tokenized into words. This led to formation of two dictionaries D_e (English) and D_b (Bengali), where each entry had the word and its corresponding frequency. Both the dictionaries had close to 100k words. It is to be noted that our hypothesis claimed that the length of the correct word and the numeronym word remains same before and after normalization. So, We take each word from the numeronym list (N_e,N_b) and find its Levenshtein distance [5] with all the words of same length, from the prepared dictionary (D_e,D_b).

1) Damerau-Levenshtein Distance: Damerau-Levenshtein Distance is defined as the least number of insertions, deletions and replacements required to convert a word to some another word. It can be implemented using dynamic programming approach.

Algorithm

$$d_{a,b}(i,j) = \min \begin{cases} 0, \text{ if } i = j = 0 \\ d_{a,b}(i-1,j) + 1, \text{ if } i > 0 \\ d_{a,b}(i,j-1) + 1, \text{ if} j > 0 \\ d_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)}, \\ \qquad \text{if } i,j > 0 \\ d_{a,b}(i-2,j-2) + 1, \text{ if } i,j > 1 \\ \qquad \text{and } a[i] = b[j-1] \\ \qquad \text{and } a[i-1] = b[j] \end{cases}$$

where $1_{(a_i \neq b_j)}$ is the indicator function equal to 0 when $a_i = b_j$ and equal to 1 otherwise. Each recursive call matches one of the cases covered by the DamerauLevenshtein distance:

- $d_{a,b}(i-1,j) + 1 d_{a,b}(i-1,j) + 1$ corresponds to a deletion (from a to b).
- $d_{a,b}(i,j-1) + 1 d_{a,b}(i,j-1) + 1$ corresponds to an insertion (from a to b).
- $d_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} d_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)}$ corresponds to a match or mismatch, depending on whether the respective symbols are the same.
- $d_{a,b}(i-2,j-2) + 1 d_{a,b}(i-2,j-2) + 1$ corresponds to a transposition between two successive symbols.

2) Workflow: The words from D_e and D_b, with minimum Levenshtein distance are extracted for each numeronym from N_e and N_b. This created a one-to-many mapping from numeronym word to dictionary words. This is shown in Figure **??**.The maximum degree of tolerance for change was kept at 2.

Primarily, we thought of extracting the correct normalized form of a numeronym, based on selecting the word with the highest frequency, from the one-to many mapping. But, selecting the most frequent word, is not a practical approach. So, to select the most probable word, we took the help of Cosine Similarity algorithm.

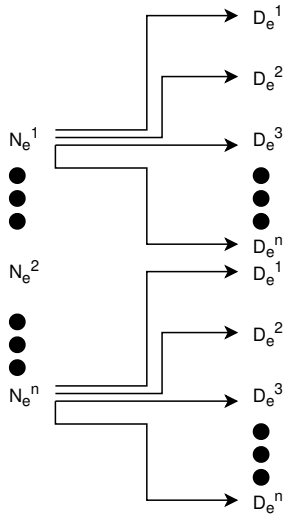

Fig. 1. Mapping of numeronyms and dictionary entries after calculating Levenshtein distance.

TABLE I
EXAMPLE OF SELECTING PROBABLE REPLACEMENT OF NUMERONYM WORD.

Input numeronym	k1n9
Most similar words	Cosine-similarity score
king	0.80
kind	0.69
kin	0.26

3) Cosine Similarity Algorithm: Cosine similarity is particularly used in positive space, where the outcome is neatly bounded in [0, 1]. The formula used in our approach is as follows.

$$Similarity = cos(\Theta) = \frac{A.B}{\|A\| \, \|B\|} = \frac{\sum_{i=1}^{n} AiBi}{\sqrt{\sum_{i=1}^{n} Ai^2} \sqrt{\sum_{i=1}^{n} Bi^2}}$$

Where A and B are the source numeronym word and one of the target dictionary words, respectively.

The word with the highest cosine similarity value was selected as the probable replacement of the numeronym. Example of numeronym replacement is shown in Table I.

B. Supervised Approach

For the supervised approach, we used a deep learning architecture that takes the numeronym words as input and generates normalized words, character by character, as output. The numeronym data-sets Nm_e (English) and Nm_b (Bengali) were merged together and it was converted to its nomralized words using manual effort. The data was then split into two parts, comprising of 9000 training data and 1000 test data. Thereafter, the numeronym and the normalized words, from the training data, were given to the deep learning architecture in character embedding. We used a time-distributed,

bi-directional LSTM hidden layer for this method and the architecture of the same is shown in Figure 2. Other parameters of the network include

- Activation: Softmax
- Optimizer: Nadam
- Loss: Sparse Categorical CrossEntropy
- Epochs: 20
- Batch Size: 64
- Validation Split: 0.1

The model returned a validation accuracy of 93.16%. When tested using 1000 test data, the model returned an accuracy figure of 89.26

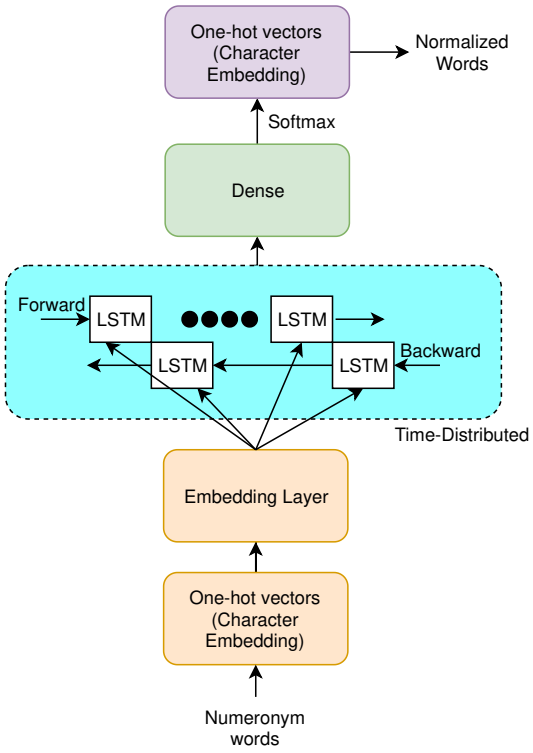

Fig. 2. Architecture of Deep Learning model.

IV. Results

We wanted to manually evaluate both our methods. So, we decided to test all the data from Nm_e (English) and Nm_b (Bengali), for the semi-supervised approach. For the supervised approach, the trained neural network model was fed with 1000 test data and consequently, it gave 1000 normalized words as output.

Since, the experiment was done on both English and Bengali numeronyms, we took the help of two linguists, who were fluent in both the languages. The linguists were asked to classify the results of our algorithm into two classes; correct and incorrect. The inter-annotator agreement for both the approaches, for both languages, are shown in Table II, III and IV, respectively.

TABLE II
Inter Annotator agreement for Semi-Supervised Approach (English).

		Linguist B	
		Correct	Incorrect
Linguist A	Correct	5563	429
	Incorrect	416	1592
Kappa		0.720	

TABLE III
Inter Annotator agreement for Semi-Supervised Approach (Bengali).

		Linguist B	
		Correct	Incorrect
Linguist A	Correct	1487	110
	Incorrect	77	326
Kappa		0.718	

TABLE IV
Inter Annotator agreement for Supervised Approach (Both English and Bengali).

		Linguist B	
		Correct	Incorrect
Linguist A	Correct	948	5
	Incorrect	3	36
Kappa		0.896	

V. Conclusion

Through this work of ours, we manage to make machines understand Numeronyms and decode them as any normal person would by looking at them. After understanding them, these systems can more accurately respond to various requirements based on Language processing. Our results are quite satisfactory in portraying the success of the algorithm and hope to find a use in near future in systems of daily needs. Lack of data-sets have led to lower accuracy which makes way for further works to be carried out based on this work. A better similarity measurement metric and selection model might give more accurate results.

References

[1] T. Arbekova, "Lexicology of the english language," *M.: High School*, 1977.

[2] V. Borisov, "Abbreviations and acronyms," *Military and scientific-technical shortenings in foreign languages./Ed. Schweitzer, AD–M.: Higher School*, 2004.

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

[4] J. L. Fleiss and J. Cohen, "The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability," *Educational and psychological measurement*, vol. 33, no. 3, pp. 613–619, 1973.

[5] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, no. 8, 1966, pp. 707–710.

[6] S. Mahata, D. Das, and S. Pal, "Wmt2016: A hybrid approach to bilingual document alignment," in *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, vol. 2, 2016, pp. 724–727.

[7] J. Ware, *Localization: For Starters*. USA: CreateSpace Independent Publishing Platform, 2016.