Journal of Information Technology and Digital World (ISSN: 2582-418X)
www.irojournals.com/itdw/

Generation and Splitting of the Compound Words in Nepali Text

Prabin Acharya[1], Subarna Shakya[2]

[1,2]Department of Electronics and Computer Engineering, Institute of Engineering, Tribhuvan University, Kathmandu, Nepal

E-mail: [1]075mscsk012.prabin@pcampus.edu.np, [2]drss@ioe.edu.np

Abstract

In Nepali language, compound word formation is mostly associated with inflection, derivation, and postposition attachment. Inflection occurs due to suffixation, whereas derivation is driven by both prefixation and suffixation. The compound word generated by the rules may produce lots of out-of-vocabulary words due to limited lexical resources and numerous exceptions. Hence, the machine learning approach can help to generate valid compounds and split them into valid morphemes that can be further used as a resource for spelling suggestions, information retrieval, and machine translation. In this research, a method to generate valid compounds from the corresponding compound splits (head word and prefix/suffix/ postpositions) is suggested. A BiLSTM based deep learning approach was used to generate and split the valid compound words. Publicly available Nepali Brihat Shabdakosh data from Nepal Academy and scraped news data were used for the experimentation. The obtained results were found to be outstanding compared to the rule-based approach applied to a similar job.

Keywords: Inflection, derivation, out-of-vocabulary word, information retrieval, BiLSTM

1. Introduction

In Nepali language, words may be formed in two ways: inflection and derivation. Inflection occurs as a result of suffixation. The inflectional suffix does not affect the meaning of the root. As a result, inflectional variations of a root generally have the same lexical category as the root. The process of generating new words from existing roots is known as derivation. In contrast to inflection, which is a process of word construction that preserves the root's lexical category, derivation can modify the root's lexical category. The close compound words are concatenated without any blank spaces and occasionally with connecting

DOI: https://doi.org/10.36548/jitdw.2022.3.007
Received: 04.08.2022, received in revised form: 22.08.2022, accepted: 09.09.2022, published: 19.09.2022

morphemes. The ability to break compound words can be a valuable resource in natural language processing, which can be further used for statistical language translation and information retrieval. Because compounding is so common, new compounds regularly appear as out-of-vocabulary (OOV) terms, which can degrade the efficacy of NLP tools.

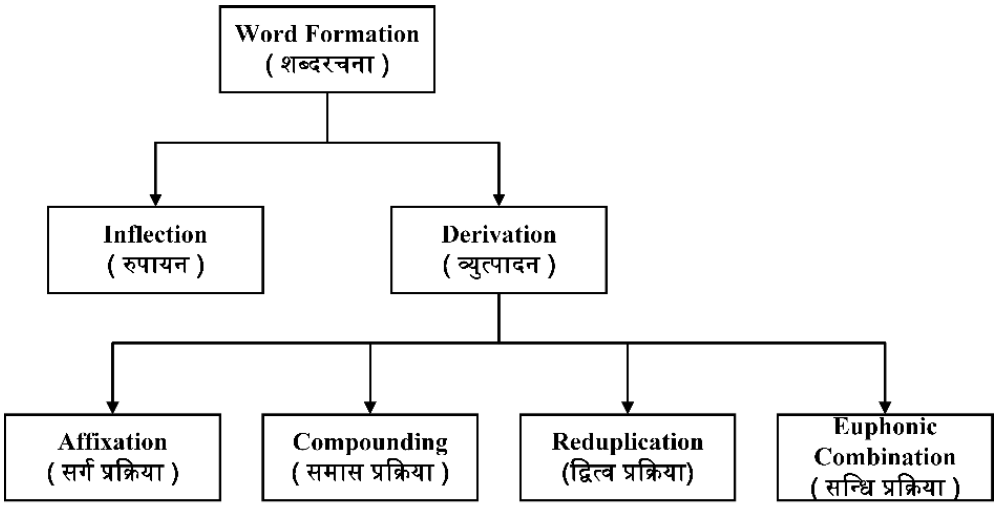

Figure 1. Word formation process in Nepali[1]

1.1 Problem Statement

Compound word formation in Nepali is governed by various grammatical rules. The compound generated by these rules can generate lots of out-of-vocabulary words. Hence, we cannot generate all relevant compound words which can be further used as a resource for machine translation or information retrieval tasks. Similarly, finding the most reasonable split in compound words is still a difficult task due to the peculiar morphology of the Nepali language. Existing splitting approaches split compound words using predefined grammatical rules, but their accuracy is low since the same compound word can be broken down in many ways to provide syntactically valid splits.

2. Related Works

There is relatively little work being done on word compound generation and split in Nepali text. However, other comparable studies in languages, particularly German and Sanskrit can be seen. For compound splits, several methods such as rule-based, statistical, and deep learning were found to be used.

[1] *https://nepalinlp.com/word-formation-and-nominal-inflections-in-nepali/*

RNNs (BiLSTM, LSTM as encoder and decoder respectively) were used to solve problems following sequence to sequence model [1]. This approach took the compound word as input and the two initial words concatenated with the '+' character as output. While in sandhi split job was divided into two stages: predicting the split point and breaking the sandhified (compound) word [1].

(Aralikatte et al., 2018) proposed a method of automatically generating split words by first learning potential split points in a compound word and implemented a deep bi-directional character RNN encoder and two attention-based decoders, seq2(seq)2 for splitting the compound word[2].

(Daðason, Mollberg, Loftsson, & Bjarnadóttir, 2020) presented a method of character-based BiLSTM model for splitting compound words[3]. (Hellwig, 2015) modeled a problem using RNN [4]. The labeling task is performed by a Recurrent Neural Network comprising an input layer, a hidden forward and backward layer, and an output layer [4]. The problem of identifying compound words from a Sanskrit document was modeled as a binary classification problem [5]. Other various approaches to compound splitting and formation are discussed in [6][7] and word structure and formation rules in Nepali grammar are discussed in [8].

3. Proposed Work

3.1 Data Description

The data collected was word-level data from Nepal Academy's Nepali Brihat Shabdakosh, which had over 60 thousand words. Besides that, Nepali news data including over 40 thousand news texts scraped from various public online news sources were used to construct the data set. These records featured compound words generated by inflection, derivation, and the attachment of postpositions.

Table 1. Raw dataset before preprocessing

S.N.	Data Source	Description	Size
1.	Nepali Brihat Shabdakosh	Existing vocabulary words with prefix/suffix and root words.	60,000+
2.	News Text (scraped)	Compound words with postpositions	40,000+

3.2 Data Preprocessing

Figure 2. Dictionary data preprocessing pipeline

The raw dictionary data comprised the compound term, its root and suffix/prefix, parts of speech, the meaning, and the index. The WORD_ID (compound word) and the POS (parts of speech containing word formation data) field was only used during data preparation, as specified in Table 2.

Table 2. Sample dictionary data

Word_ID	Word	POS	Meaning	Index
अऋणी	अऋणी	वि. [सं. अ+ऋणी]	ऋण नभएको; रिन तिरि सकेको; ऋणमुक्त; उऋण। विप.ऋणी।	72
जर्मराइ	जर्मराइ	ना.[Öजर्मराउ (+आइ)]	जर्मराउने भाव,क्रिया वा प्रक्रिया।	19390
थ्याप्चिनु	थ्याप्चिनु	अ.क्रि.[थ्याप्चो+इ+नु]	थ्याप्चो हुनु;थेप्चिनु।	25841
माधिमे	माधिमे	ना. [सं. मा+नेवा.+धिमे]	नाङ्गै हात र बेतको घुमेको गजाले बजाइने, धुन्या बाजा समूहको मुख्य तालबाजा;	44469
स्वाँस्वाँ	स्वाँस्वाँ	क्रि.वि. [अ.मू. स्वाँ (द्वि.)]	१. परिश्रम वा दमले छिटछिटो र लामो सास चल्ने गरी; तिखो स्याँस्याँ।।	58323

Various data cleaning steps were included in the dictionary data preprocessing as shown in Figure 2. The obtained data after word filter from raw dictionary data were transformed into compound words and the corresponding splits. Unwanted symbols, special characters, and Devanagari numbers were eliminated. This process entailed manual normalization and the elimination of irrelevant information to qualify the data for model training input as shown in Table 3.

The raw news data comprised of articles covering a wide range of texts from a domain like finance, politics, sports, culture etc. considerable number of incorrect/invalid words were also present along with valid compound words. The challenge of identifying and

validating compound words, as well as pre-processing raw text data to build a dataset, was also essential.

Table 3. Data normalization and removal through manual validation

Intermediate data	**Preprocessed**	**Action**
जिङ्ग्रङ्+ग/ङ => जिङ्ग्रङ्ग/जिङ्ङ्रङ्ङ	जिङ्ग्रङ्+ग=>जिङ्ग्रङ्ग जिङ्ग्रङ्+ङ=> जिङ्ङ्रङ्ङ	Normalized
टप्>टपक्+क => टपक्क	टपक्+क => टपक्क	Normalized
बराजु =>ना. [बडा+ज्यू< सं. वरा–आर्य]	बराजु => बडा+ज्यू	Normalized
जय+नेपाल => जय नेपाल	open compound	Removed
जाउ+अत+आउ+अत => जावतआवत	more than one morphemes	Removed
सं+विच्छादन => बिछ्यौना	incomplete data	Removed
जुरुङजुरुङ => ना.[अ. मू. जुरुङ(द्वि.)]	incomplete data	Removed

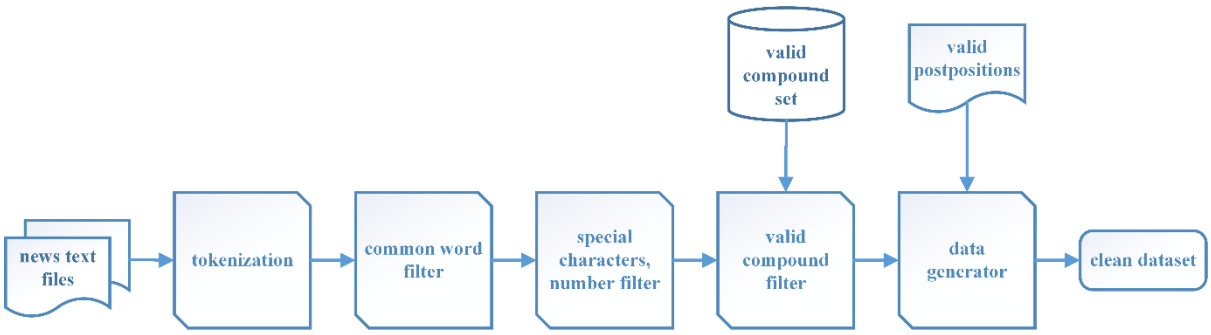

Figure 3. News data preprocessing pipeline

The pre-processing of news data included many phases of data cleaning and sanitization. News texts were used to produce data with suffixed postpositions. The prefixed compound terms were not included in this dataset. The scraped news texts were initially passed to the tokenizer, which converted the texts into a set of words. The common words were excluded from the list of tokenized words, and unique terms were retrieved. Following that, special characters were removed from the list of words, which was then intersected with the list of valid dictionary terms to get the words with valid postpositions. Finally, the compound words were broken down using the set of valid postpositions to get the final dataset. The pre-processing steps can be seen in Figure 3. An example of the dictionary data preprocessing steps is shown on the Figure 4.

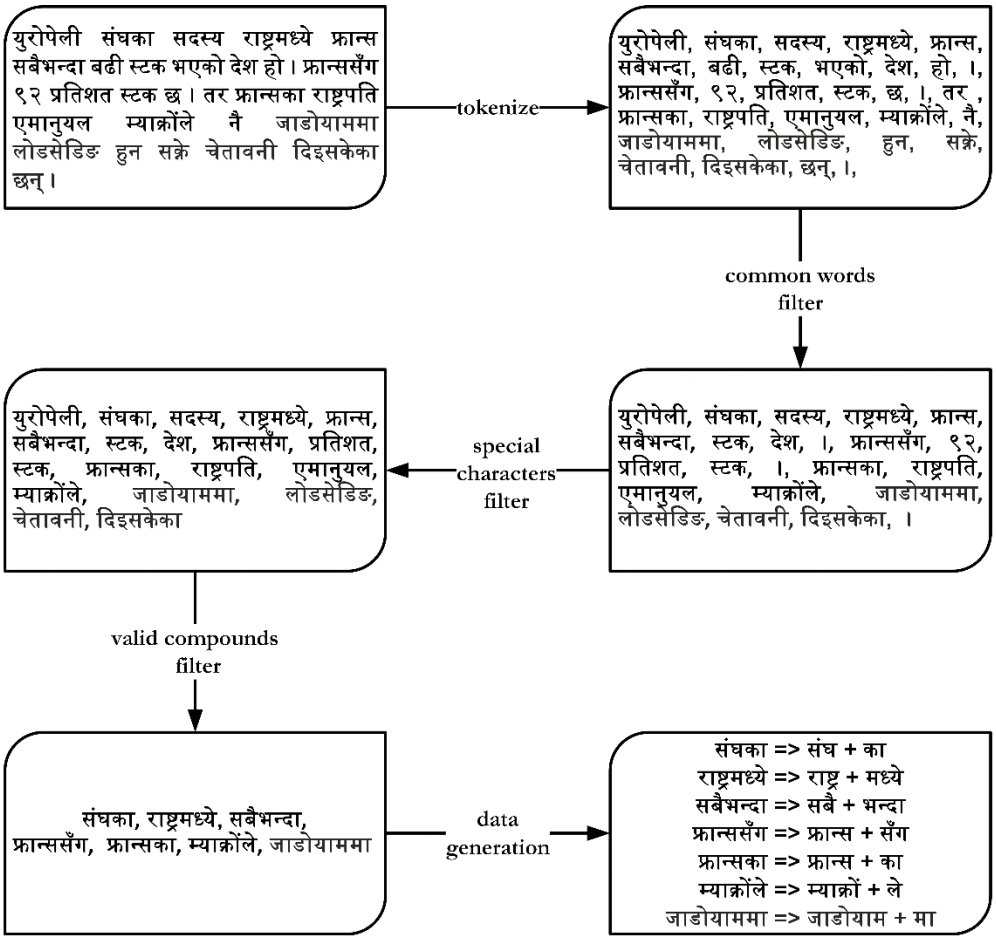

Figure 4. News text data preprocessing example

3.3 Compound Generation

The compound generation task has been modeled as the sequence-to-sequence prediction task. A provided sequence of the words (containing head/root and modifier/suffix/prefix/postpositions) ultimately generates another sequence (compound word). The example of the compound formation sequence can be seen in the example below.

बोको+ए => बोके

घ्याच्+च => घ्याच्च

असन्तुष्ट+हरूले => असन्तुष्टहरूले

Instead of relying on external lexical resources, this method learns to incorporate word category-specific rules. As a result, we must occasionally concatenate words to generate new words that comprise omitted or additional characters from component words. Here for compound generation, we used Bi-LSTM/LSTM based encoder-decoder model. The detailed block diagram of the compound generation process is shown in Figure 5.

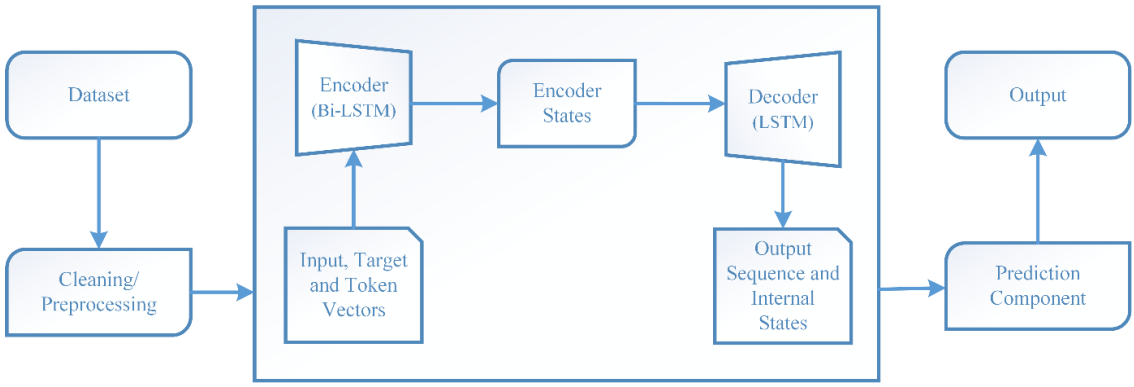

Figure 5. Block diagram of the compound generation process

3.4 Compound Split

The block diagram for the compound split process is shown in Figure 6. Many split points may be found in a single compound word, predicting the optimum split point is challenging, a two-stage A double decoder approach is required in which The first one predicts the optimum split point and the second one splits the compound on the given window.

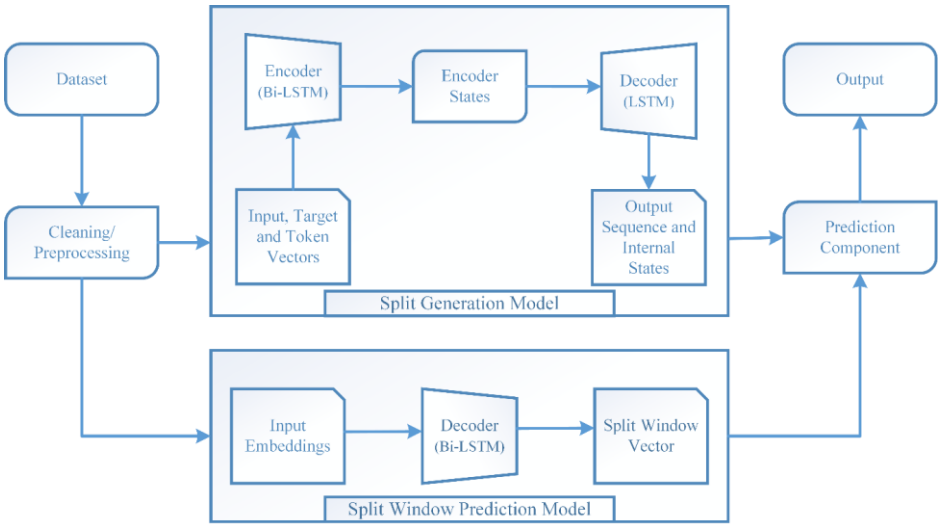

Figure 6. Block diagram of the compound split process

In the compound window estimation task, the compound word is the input sequence and the target output is an integer array with the same number of items as the characters in the compound word. Except for the values corresponding to the compound-window characters, which are set to 1, all items in this array is 0. The RNN model used in this the approach is intended to output an array with a size equal to the length of the input compound word. Except for the items at the compound-window position, all elements in this array must be zero.

उनिहरुकोभान्दा => [0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0]

In the above example, **को** is treated as the compound window whose characters are set as 1 whereas other input characters are set as 0. So the compound will split at **को**.

For compound window prediction RNN was employed as the basic RNN cell for this decoder, with bidirectional LSTM cells providing somewhat better overall performance than regular RNN cells. At each time step, the output vector was linked to a dense layer with an output array of unit length. The output array member corresponding to the input character is represented by this single output value. Input character sequences were encoded using one-hot vectors. The compound split algorithm after split window prediction is shown below.

Algorithm: Compound Split

Require: *RNN model, estimate the compound-window*

S_1 , S_2 → *two words generated by seq2seq model on compound-window*

N_1 → *previous section of the compound word's first expected split* P_1

N_2 → *continuation of the compound word's second expected split* P_2

$P_1 \rightarrow N_1 + S_1$

$P_2 \rightarrow S_2 + N_2$

4. Results and Discussion

The experiments were carried out on a Google Colab. A new class-balanced dataset was also created and experimented with by blending the two datasets discussed in Section 3.2. The model and test results were evaluated using the K-Fold cross validation method.

Table 4. Training attributes of different datasets for compound generation and split task

Attributes/Dataset	Tasks					
	Compound Generation			Compound Split		
	Dictionary Data	*News Data*	*Combined Data*	*Dictionary Data*	*News Data*	*Combined Data*
Training samples	5992	9115	15106	3717	7347	10369
Unique tokens	51	51	51	51	51	51
Max. Input sequence	7	7	7	5	5	5
Max. output sequence	9	8	9	11	8	11

For all of the datasets, data were split into training and test set 80/20 (80% training data and 20% test data). The test was carried out on a separate test dataset as well as the test set generated from the same dataset. Table 4 exhibits attributes of the training data (on different datasets) whereas Table 5 shows the model parameters for training.

Table 5. Training parameters for different datasets in compound generation and split task

Parameters/Dataset	Tasks					
	Compound Generation			Compound Split		
	Dictionary Data	*News Data*	*Combined Data*	*Dictionary Data*	*News Data*	*Combined Data*
Batch size	64	64	64	64	64	64
Number of epochs	100	40	100	100	40	100
Latent dimensionality of the encoding space	16	16	16	128	128	128

K-Fold cross validation with k=5 was used to evaluate the models on different datasets for a compound generation task. For each dataset, the model was trained for five folds and the accuracy of the model was tested for each fold on the corresponding test dataset. The exact matching of the complete compound phrase was used in evaluating the accuracy of the model. For compound formation even if the model correctly concatenates across word boundaries, an error in a character before or after the generated word is considered a failure.

The aforementioned result was compared to the compound term found in our news and dictionary test data. This model's success was based on training data that covers as many rules as possible and the amount of noise and bias in data. Table 6 demonstrates the test accuracy at each fold and average accuracy for compound formation. The bar graph is shown in Figure 6 exhibits training vs validation vs test accuracy at each fold for the dictionary dataset, news dataset and combined dataset.

The model's accuracy varied among datasets, as seen in Figure 7. The amount of bias in a distinct dataset was responsible for this variation. This variance lead to the elimination and addition of several characters during compound formation, making learning data for the model challenging. As a result, the test accuracy on the dictionary dataset was lower (77.65%) then on the news text dataset (97.27%) and combined dataset (86.60%)

Table 6. Test accuracy of compound generation model for different datasets over each folds

Fold\Dataset	Dictionary Dataset	News Dataset	Combined Dataset
Fold 1	76.44 %	96.92 %	86.20%
Fold 2	77.65 %	97.27 %	85.60%
Fold 3	74.58 %	94.21 %	87.52%
Fold 4	77.65 %	96.05 %	85.56%
Fold 5	77.17 %	97.01 %	86.22%

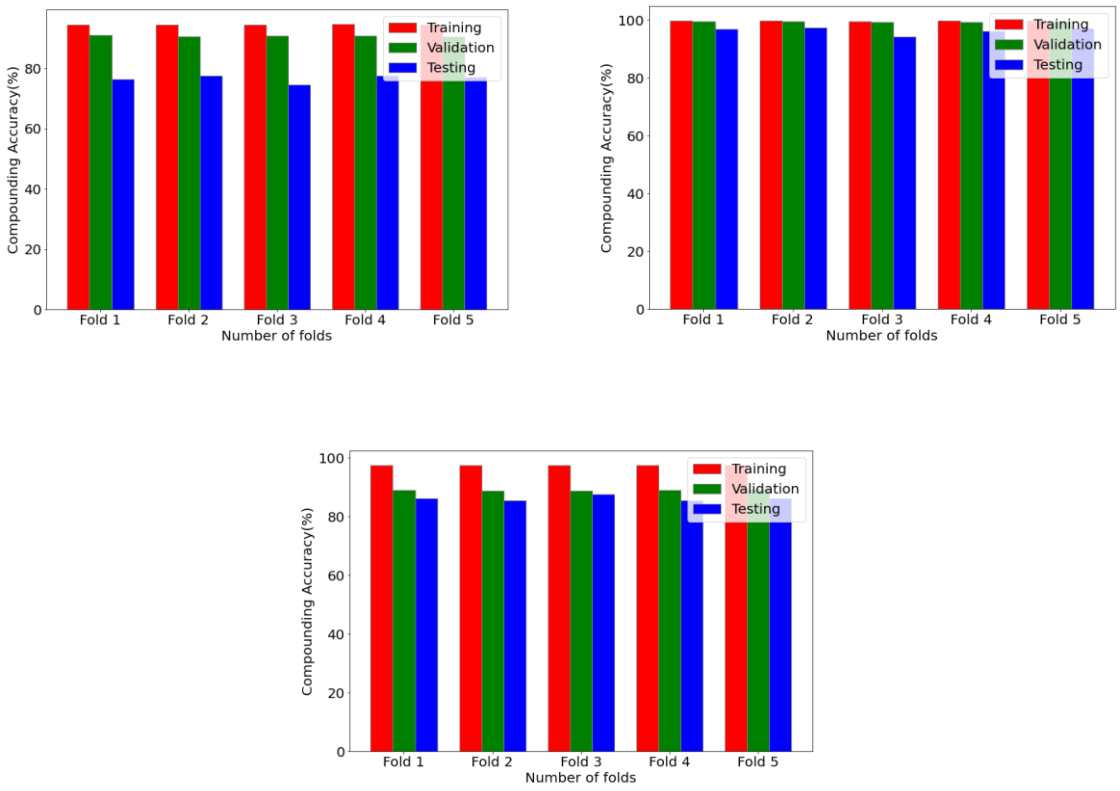

Figure 7. Training vs validation vs test accuracy comparison for compound generation on dictionary, news and combined dataset respectively

K-fold cross validation was used to find the best fit model exactly as it was employed in the compound formation process. Due to data size constraints, only k=4 (four folds) was used in this case. The generated sets of splits were precisely matched with the expected set of splits for an input compound. Even a single character omission or addition in any of the ensuing splits was considered a failing case.

Table 7. Test accuracy of compound split model for different datasets over each folds

Fold\Dataset	Dictionary Dataset	News Dataset	Combined Dataset
Fold 1	75.09 %	97.72 %	84.14%
Fold 2	73.59 %	97.29 %	85.28%
Fold 3	74.25 %	97.93 %	83.96 %
Fold 4	75.80 %	94.06 %	86.51 %

Compared to the compound generation task, the accuracy variation for the compound splitting task was nearly identical as in Figure 8. The dictionary data set's test accuracy (75.8%) was lower than that of the news text dataset and combined dataset (97.93% and 86.51%, respectively). The difference in model accuracy is related to noise and diversity present in the dictionary dataset compared to the other two datasets.

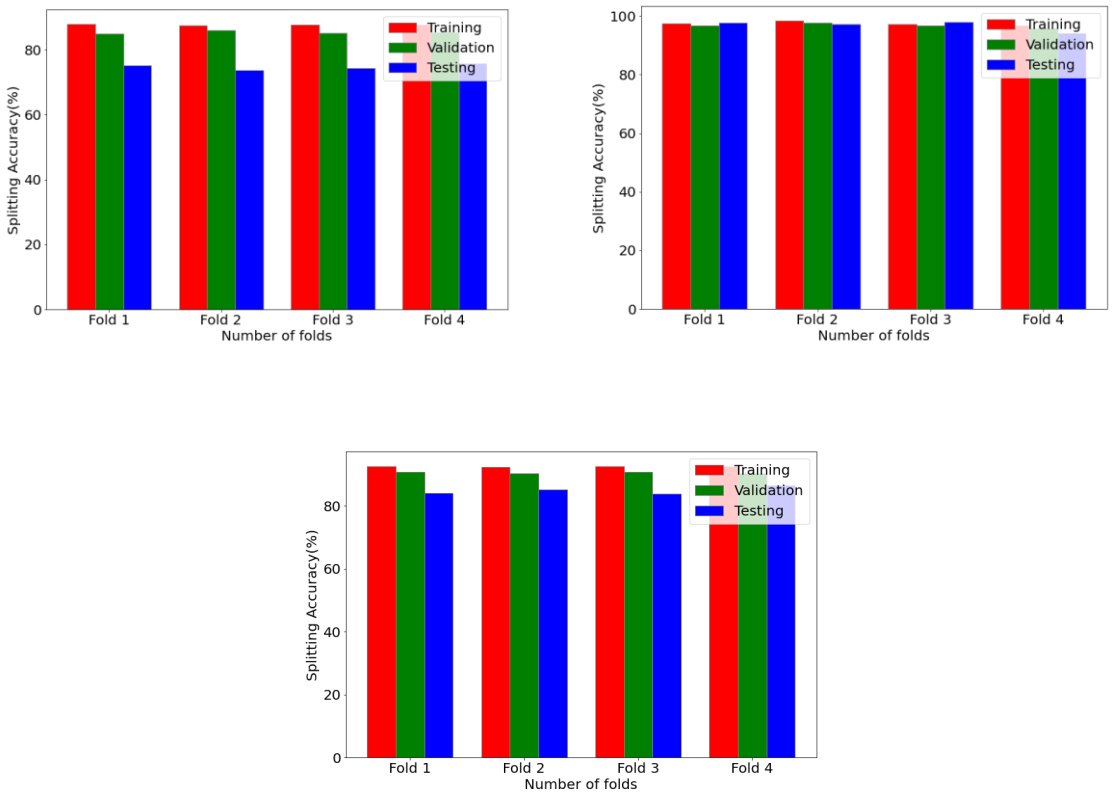

Figure 8. Training vs validation vs test accuracy comparison for compound generation on dictionary, news and combined dataset respectively

In compound split task, it was observed that certain compounds are syntactically and semantically correct but did not match the intended splits, resulting in a failed case. An example of such a case is shown below.

अडिरहेकी	अडि+रहेकी	अडिरहे+की
Compound	**Generated Split**	**Actual Split**

Both the generated split and the actual split are valid cases but our test scenario failed the above case due to the exact match constraint, hence the accuracy in the news dataset can be improved if such cases are deemed valid cases.

	Passed Results			Failed Results		
Compound Generation	Split	Generated	Actual	Split	Generated	Actual
	सुरो+ई	सुरी	सुरी	जोड+नु	जोड्नु	जोर्नु
	घ्याच्+च	घ्याच्च	घ्याच्च	दंश्+नु	दस्नु	डस्नु
	अराउ+आइ	अराइ	अराइ	ख्याक्+क	ख्याक्क	ख्वाक्क
	असर+सम्मका	असरसम्मका	सरसम्मका	मङ्गल+बारेकी	मङ्गलबारेका	मङ्गलबारेकी
	असन्तुष्ट+हरूले	असन्तुष्टहरूले	असन्तुष्टहरूले	आँध्र+पछि	आँध्रीपटि	आँध्रीपछि
	अण्डा+देखि	अण्डादेखि	अण्डादेखि	अध्येता+हरूले	अध्याएतहरूकै	अध्येताहरूले
Compound Splitting	Compound	Generated	Actual	Compound	Generated	Actual
	भ्वाँक	भ्वाँक्+अ	भ्वाँक्+अ	सुकाउरो	सुक+आउरो	सुक्+आउरो
	टुक्कुक्क	टुक्कुक्+क	टुक्कुक्+क	थिच्याइ	थिरि+याइ	थिर्+याइ
	दबाइ	दब्+आइ	दब्+आइ	खिप्नु	छुप्+नु	क्षिप्+नु
	आधुनिकताबीचको	आधुनिकता+बीचको	आधुनिकता+बीचको	अटोप्रति	अट्टो+पति	अटो+प्रति
	इनामवाला	इनाम+वाला	इनाम+वाला	अडिरहेकी	अडि+रहेकी	अडिरहे+की
	गर्नुअघिको	गर्नु+अघिको	गर्नु+अघिको	अरूद्वारा	अरी+द्वारा	अरू+द्वारा

Figure 9. Model output vs expected output for compound generation and split task

Figure 9 shows the comparison between the output generated by the model and the actual result to be generated (expected output) for the dictionary dataset and news dataset. From the result, of compound generation model we can analyse that most of the successful outputs are the ones that have less omission or addition during compound word formation while failed cases in the dictionary dataset include mostly the constituent root word from other languages like Sanskrit, Hindi, Persian, etc. than Nepali (दंश् (Sanskrit) + नु => गाँस्नु). Due to the existence of such data the model is not performing well in the dictionary dataset. In addition, the occurrence of homonyms and their word split in the dataset affected accuracy. In the case of the News dataset, the failed cases are mostly due to single character transformation which can be optimized to obtain maximum accuracy.

Similarly, in the case of compound split model results, it was observed that some of the failed outputs were the ones where the wrong suffix/prefix or postpositions are generated

with the correct root words or wrong root words were generated along with the right suffix/prefix and postpositions (सुकाउरो => सुक+आउरो, सुक is the wrong headword in this case which should be सुक् but सुक is also a valid headword). These issues can be resolved with another level of manual validation and removal of noise creating data from the dataset.

5. Conclusion

The applied methodology provides a mechanism for generating and splitting complex words without the usage of any external resources such as language models or morphological or phonetic analyzers. This method is effective in terms of computational resources and implementation. This research work can be refined further with the addition of sufficient training data representing optimum rules and exploring strategies to eliminate biases in current training data.

Acknowledgement

We'd like to extend our sincerest gratitude to the Department of Electronics and Computer Engineering, Pulchowk Campus for providing continuous support throughout the research. We would also like to thank all teachers and colleagues for their direct and indirect help related to research.

References

[1] Dave, Sushant, Arun Kumar Singh, Dr Prathosh AP, and Prof Brejesh Lall. "Neural compound-word (Sandhi) generation and splitting in Sanskrit language." In *8th ACM IKDD CODS and 26th COMAD*, pp. 171-177. 2021.

[2] Aralikatte, Rahul, Neelamadhav Gantayat, Naveen Panwar, Anush Sankaran, and Senthil Mani. "Sanskrit Sandhi Splitting using seq2 (seq)^ 2." *arXiv preprint arXiv:1801.00428* (2018).

[3] Daðason, Jón Friðrik, David Erik Mollberg, Hrafn Loftsson, and Kristín Bjarnadóttir. "Kvistur 2.0: a BiLSTM Compound Splitter for Icelandic." *arXiv preprint arXiv:2004.07776* (2020).

[4] Hellwig, Oliver. "Using Recurrent Neural Networks for joint compound splitting and Sandhi resolution in Sanskrit." In *4th Biennial Workshop on Less-Resourced Languages*. 2015.

[5] Premjith, B., Chandni Chandran, Shriganesh Bhat, Soman Kp, and P. Prabaharan. "A machine learning approach for identifying compound words from a Sanskrit text." In *Proceedings of the 6th International Sanskrit Computational Linguistics Symposium*, pp. 45-51. 2019.

[6] Stymne, Sara, Nicola Cancedda, and Lars Ahrenberg. "Generation of compound words in statistical machine translation into compounding languages." *Computational Linguistics* 39, no. 4 (2013): 1067-1108.

[7] Le, Yvonne. "Choosing the most reasonable split of a compound word using Wikipedia." (2017).

[8] Bal, Bal Krishna. "Structure of Nepali grammar." *PAN Localization, Madan Puraskar Pustakalaya, Kathmandu, Nepal* (2004): 332-396.

Author's biography

Prabin Acharya currently pursuing his MSc. Degree in Computer System and Knowledge Engineering from Pulchowk Campus, Institute of Engineering, Tribhuvan University.

Subarna Shakya received MSc and PhD degrees in Computer Engineering from the Lviv Polytechnic National University, Ukraine, in 1996 and 2000, respectively. Currently, he is the Professor at the Department of Electronics and Computer Engineering, Pulchowk Campus, Institute of Engineering, Tribhuvan University. His research interest includes e-government system, distributed system, cloud computing, and software engineering and information system, Deep Learning, and Data Science.