Anuj@DPIL-FIRE2016: A Novel Paraphrase Detection Method in Hindi Language using Machine Learning

Anuj Saini
Sapient Global Markets
Gurgaon, Haryana
asaini13@sapient.com

ABSTRACT

Every language possesses plausible several interpretations. With the evolution of web, smart devices and social media it has become a challenging task to identify these syntactic or semantic ambiguities. In Natural Language Processing, two statements written using different words having same meaning is termed as paraphrasing. At FIRE 2016, we have worked upon the problem of detecting paraphrases for the given Shared Task DPIL (Detecting Paraphrases in Indian Languages) in Hindi Language specifically. This paper proposed a novel approach to identify if two statements are paraphrased or not using various machine learning algorithms like Random Forest, Support Vector Machine, Gradient Boosting and Gaussian Naïve Bayes on the given training data set of two subtasks. In cross validation experiments, Random Forest leads the other methods in terms of F1-score. The experimental results depicts that our algorithm gives better performance with the ensemble learning method than individual approaches for such classification problem. This can be used in various applications such as question-answering system, document clustering, machine translation, text summarization, plagiarism detection and many more.

CCS Concepts

• **Theory of computation →Random Forest**
• **Computing methodologies→Natural language Processing**

Keywords

Paraphrase detection; Machine Learning; Natural Language Processing; Soundex; Semantic Similarity; Random Forest

1. INTRODUCTION

With the plethora of information generated by the web these days, it is challenging to understand the semantics of different languages when each language has its own scripts and linguistic rules. Hindi language is still in its early stage concerning to natural language processing and applications [1]. Currently significant amount of research work has already been done for English language but there is a huge scope for Hindi language. Paraphrases are sentences or phrases which conveys the same meaning using different words [2]. Paraphrase detection is an important building block in Natural Language Processing (NLP) pipeline [8]. Previously, many researchers have investigated ways of automatically detecting paraphrases on formal texts [3].Various state-of-the-art paraphrase identification techniques have been summarized in an excellent manner by ACL [4]. The objective of our work is motivated by the shared task of DPIL [11] organized by the Forum for Information Retrieval Evaluation (FIRE 2016). There were two subtasks given for the classification problem. Subtask 1 is to classify two Hindi sentences into two classes: P (Paraphrased) or NP (Non-Paraphrased). Subtask 2 is to classify them into three classes which are P (Paraphrased), SP (Semi-Paraphrased) or NP (Non-Paraphrased). For detecting paraphrases, it is very important to understand the language at an initial stage. Starting from tokenization, stemming, lemmatization, stop words, phonetics and POS tags etc. need to be identified before comparing two texts. And as machine learning algorithms works on numeric data, so it's important to convert our textual data into corresponding numbers known as text vectors. A vector denotes the numerical representation of text comparison of two sentences. We have generated a set of vectors for each data point using first two steps of our proposed approach and trained the model in the third step to get the results.

The paper has been organized as follows. Section 2 gives the description of the training dataset provided by the organizing committee. Section 3 presents the proposed approach for paraphrase detection. The experimental evaluation has been carried out in Section 4. Section 5 concludes our research work followed by acknowledgment and references given in Section 6 and Section 7 respectively.

2. DATA SET DESCRIPTION

There were a total of 2500 data points in the training set with 2 classes P and NP to be used for classification in Subtask1, whereas Subtask 2 had 3 classes and a total of 3500 data points. A detail distribution of classes and its count for both the tasks is mentioned in Table1.

Table 1. Classes & its count for SubTask1 and SubTask2

Class	SubTask1	SubTask2
P	1500	1500
NP	1000	1000
SP	-	1000

Each data point contains a pID, a unique id for each data point and two Hindi sentences and their final tagged Class. The initial analysis of the data surfaced some noise, majorly in NP class. A few examples of false negatives and false positives have been identified and listed down in Table 2. Here the examples for class NP and class P denotes false negatives and false positives respectively.

Table 2. False Negatives and False Positives in the dataset

pID	Sentence1	Sentence2	Class

HIN2802	रविशंकर ने कहा मुझे लगता है कि आईएसआईएस कोई पीस टॉक नहीं चाहता	मुझे लगता है कि आईएसआईएस कोई पीस टॉक नहीं चाहता: रविशंकर	NP
HIN2852	पाक के फॉरेन सेक्रेटरी एजाज अहमद चोधरी मंगलवार को हार्ट ऑफ एशिया के ऑफिशियल्स की बैठक में भाग लेने के लिए भारत पहुंच रहे हैं	पाकिस्तान के विदेश सचिव एजाज अहमद चोधरी हैदराबाद हाउस में हार्ट ऑफ एशिया सम्मेलन में भाग लेंगे	NP
HIN2958	इस किताब में कांग्रेस उपाध्यक्ष राहुल गांधी को करिश्माई नेता भी बताया गया है	इस किताब में कांग्रेस उपाध्यक्ष राहुल गांधी को करिश्माई नेता भी बताया गया और इसको लेकर लोकसभा में बुधवार को सत्तापक्ष और विपक्ष के बीच काफी नोकझोंक हुई	NP
HIN3032	बॉलीवुड अभिनेत्री बिपाशा बसु ने शनिवार को एक निजी समारोह में करण सिंह ग्रोवर के साथ शादी रचाई	बॉलीवुड एक्ट्रेस बिपाशा बसु और करण सिंह ग्रोवर ने शनिवार को मुंबई में की शादी	NP
HIN0230	मुद्राकोष में बढ़ेगा भारत का रुतबा	ऐएमएफ के दस बड़े सदस्य देशों में भारत भी शामिल	P

We have accepted this noise present in the given training data without any filtering of such misclassified records and trained model on all the data points.

3. THE PROPOSED METHOD

The proposed approach includes three major steps which are text preprocessing, feature generation and classification model. Text preprocessing is done in various steps such as tokenization, stemming, soundex, stop word removal and handling synonyms. Feature generation involves the creation of five new features which will be used as an input to the classifier for classification of paraphrases. Classification model includes the model training using the four machine learning algorithms. All the steps are described below.

3.1 Text Preprocessing

For each data point, the preprocessing steps are as follows:

3.1.1 Text Encoding

We have encoded the sentences using standard UTF-8 encoding that handles scripting of Hindi language.

3.1.2 Tokenization

We have tokenized the sentences into words using NLTK library.

3.1.3 Phonetics Transformation

We have applied custom set of rules for phonetics. The normalization of phonetics has been done using soundex algorithm [9]. It looks for specific characters and replaces them with their corresponding metaphor characters. For example,

न् □ न or ज़ □ ज

3.1.4 Tokens Stemming

We have applied some more set of rules for stemming into its basic form. A set of Hindi suffixes characters were removed to get normalized Hindi word. For example,

[◌ो",◌े",◌ू",◌ु",◌ी",◌ि",◌ा"][कर",◌ाओ",◌िए",◌ाई",◌ाए",ने",नी",ना",ते",◌ीं",ती",ता",◌ाँ",◌ां",◌ों",◌ें"]

[◌ाकर",◌ाइए",◌ाईं",◌ाया",◌ेगी",◌ेगा",◌ोगी",◌ोगे",◌ाने",◌ाना",◌ाते",◌ाती",◌ाता",तीं",◌ाओं",◌ाएं",◌ुओं",◌ुएं",◌ुआं"]

3.1.5 Stop Words Filtering

We have removed irrelevant words from the sentences using a standard list of 164 Hindi Stop words. For example,

[सारा, से, सो, संग, ही, हुआ, हुई, हुए]

3.1.6 Synonyms Expansion

We have used Hindi WordNet, an extensive lexical dictionary of Hindi language having 40K~ synsets, to fetch synonyms for Hindi words. It was developed by researchers at the Center for Indian Language Technology, Computer Science and Engineering Department, IIT Bombay and we have downloaded it from the mentioned link in [5].

All the text preprocessing steps have been summarized with examples in the below Table 3.

Table 3. Text Preprocessing Steps

PreProcessing	Input	Preprocessed
Tokenization	कपिल शर्मा फोर्ब्स	[कपिल, शर्मा, फोर्ब्स]
Soundex	हजअर लोट	हज़अर लौट
Stemming	अनियमितताओं, दिल्ली	अनियमित, दिल्ल
Stop Words Removal	काफी निराश था और ड्रेसिंग रूम में लोटते हुए रोने लगा	काफी निराश ड्रेसिंग रूम लोटते हुए रोने लगा
Synonyms	इंडिया आखिरी	भारत फाइनल

3.2 Feature Generation

After the preprocessing of sentences following features have been generated in the form of vectors to be passed as an input to the classifier.

3.2.1 *Common Tokens*

The number of common tokens amongst two sentences is used as a feature. These tokens have been generated by comparing the preprocessed tokens after removing the stop words and then taking intersection of them symbolized as follows.

Tokens (sentence1) ∩ Tokens (sentence2)

3.2.2 *Normalized common Tokens*

We have normalized the common tokens generated in the first feature to compute the proportion of commonality of tokens between two sentences. The value will be in the range of 0 and 1.It is 0 when there are no common tokens between two sentences and 1 if all tokens between the two sentences are exactly the same. Mathematically, It has been calculated by dividing the common tokens by number of unique tokens in both the sentences as shown below.

Common tokens/Unique Tokens (sent1, sent2)

3.2.3 *Common IDF Score*

Sum of IDF scores of common tokens from two sentences is used as numeric similarity vector.

ΣIDF score (common tokens)

IDF (Inverse Document Frequency) is defined as inverse of document frequency which is used to identify the importance of a token in a given corpus. Represented as:

$$\mathrm{idf}_t = \log \frac{N}{\mathrm{df}_t}.$$

Here a document is an individual sentence. At first IDF has been calculated for all tokens and kept for reference. Then during feature generation process, IDF of common tokens has been calculated and is used as a feature.

3.2.4 *Normalized Common IDF Score*

Here the proportion of IDF score of common tokens between two sentences is computed for normalization and used as a vector to model. It gives us normalized common IDF score ranges between 0 to 1 of common tokens. It can be calculated as follows.

IDF of Common tokens/

Total IDF of Unique tokens (sent1, sent2)

3.2.5 *Sentences Length*

It denotes the count of number of tokens in sentence 1 and sentence 2 as separate columns.

3.3 Classification Model

Features generated in section 3.2 have been used as training data to train the classifiers using Python Scikit library. Here, we have used four popular machine learning algorithms which are random forest, support vector machine and gradient boosting and compared their performance. Since, random forest being an ensemble learning method outperforms the other individual methods, is implemented by growing many classification trees and having them "vote" for a final decision according to a majority role [6].We have focused on tuning its hyper parameters to enhance the predictive ability of the model. Key parameters are as follows:

3.3.1 *n_estimators*

These are the number of trees that we want to build before having the vote for the final decision. More the number of trees better the performance however it also increases the time complexity.

3.3.2 *max_depth*

It is the maximum depth of the tree which needs to be tuned.

3.3.3 *min_samples_leaf*

These are the minimum number of samples or observations required in a terminal node of the tree.

3.3.4 *min_samples_split*

These are the minimum number of samples or observations needed in a node to be considered for splitting.[7]

We have selected the best set of hyper parameters for Random Forest using Grid Search of Scikit which resulted in the following values n_estimators - 500, max_depth - 10, min_samples_leaf - 4 and min_samples_split – 4 and trained our training data. Overall training time for model is less than 1 second on quad-core Machine with 8GB of RAM.

4. EXPERIMENTAL RESULTS

We have used 10 fold cross validation to compute overall accuracy for the system. In this work three evaluation metrics have been considered which are precision, recall and f1-score. We have calculated the values of the evaluation parameters for all of the four respective algorithms. For the subtask 1 we have got the overall accuracy of 0.92 with F1 score of 0.94 maximum for Random Forest algorithm. Detailed performance matrix of the model is given as below in Table 4 in which we have to predict for 2 classes.

Table 4. Subtask 1 Scores Summary

Class	Precision	Recall	F1-Score	Algorithm
P	0.94	0.93	0.94	Random Forest
NP	0.90	0.91	0.90	
Avg/Total	0.92	0.92	0.92	
NP	0.93	0.91	0.92	Gradient Boosting
P	0.87	0.89	0.88	
Avg/Total	0.9	0.9	0.9	
NP	0.92	0.84	0.88	SVM
P	0.79	0.89	0.84	
Avg/Total	0.87	0.86	0.86	
NP	0.92	0.94	0.93	Gaussian Naïve Bayes
P	0.91	0.88	0.9	
Avg/Total	0.92	0.92	0.92	

Subtask 2 which had same problem with 3 classes to predict from. We have used similar approach and similar features set for training our model. With 3 classes and larger training set of 3500 data points we have got overall accuracy of 0.85 with 10 cross folds and a F1 score of 0.91 which is again maximum for Random forest algorithm. Detailed summary of performance matrix of subtask2 is given in Table 5.

Table 5. Subtask 2 Scores Summary

Class	Precision	Recall	F1-Score	Algorithm
NP	0.90	0.91	0.91	Random Forest
P	0.81	0.80	0.80	
SP	0.83	0.82	0.82	
Avg/Total	0.85	0.85	0.85	
NP	0.89	0.90	0.89	Gradient Boosting
P	0.79	0.80	0.79	
SP	0.84	0.81	0.83	
Avg/Total	0.85	0.85	0.85	
NP	0.89	0.82	0.86	SVM
P	0.74	0.67	0.70	
SP	0.68	0.82	0.74	
Avg/Total	0.79	0.78	0.78	
NP	0.87	0.93	0.9	Gaussian Naïve Bayes
P	0.68	0.73	0.71	
SP	0.76	0.62	0.68	
Avg/Total	0.78	0.79	0.78	

Following figures gives the summarized view of the performance of the various machine learning algorithms for both subtasks

Figure 1: Subtask 1 Results Comparison

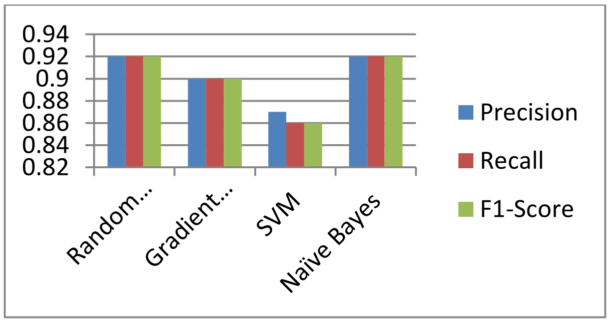
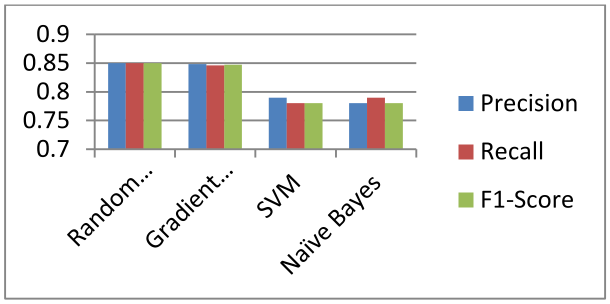


Figure 2: Subtask 2 Results Comparison



5. CONCLUSION

In this paper we have proposed our novel approach for the detection of Hindi paraphrases which is a very important building block of semantic text analysis. Building a question answering system, document clustering, knowledge extraction, plagiarism detection, building ontologies etc. are the potential applications for paraphrase identification in NLP [10]. After comparing all the four machine learning algorithms used in our model, random forest is giving best results with F1 score of 0.94 for subtask1 and 0.91 for subtask2 which can be further improved by using more robust phonetics and synonyms replacements. One limitation of our research work is that we have not removed outliers from the training data which could slightly improve the system performance. In our future work we will include Part of Speech tagging in feature generation which plays an important role in paraphrase detection, as nouns and verbs are key elements for paraphrasing.

6. ACKNOWLEDGMENTS

We would like to thank organizers for conducting this shared task and also building the training data. We also would like to thank Sapient Corporation for giving us an opportunity to work and explore the world of text analytics.

7. REFERENCES

[1] Sethi, N., Agrawal, P., Madaan, V., and Singh, S.K. July 2016. A Novel Approach to Paraphrase Hindi Sentences using Natural Language Processing. *Indian Journal of Science and Technology*, Vol 9(28), DOI: 10.17485/ijst/2016/v9i28/98374.

[2] Kumar, N. 2014. A Graph Based Automatic Plagiarism DetectionTechnique to Handle Artificial Word Reordering and Paraphrasing. A. Gelbukh (Ed.): *CICLing* 2014, Part II, LNCS 8404, pp. 481–494, Springer-Verlag Berlin Heidelberg 2014.

[3] Xu, W., Callison-Burch, C., and Dolan, W. B. 2015. SemEval-2015 Task 1: Paraphrase and Semantic Similarity in Twitter (PIT*), Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 1–11, Denver, Colorado, June 4-5, Association for Computational Linguistics

[4] https://www.aclweb.org/aclwiki/index.php?title=Paraphrase_Identification_(State_of_the_art)

[5] http://www.cfilt.iitb.ac.in/wordnet/webhwn/downloaderInfo.php

[6] Zhang, W., Zeng, F., Wu, X., Zhang, X., and Jiang, R.A. 2009. comparative study of ensemble learning approaches in the classification of breast cancer metastasis, *International Joint Conference on Bioinformatics, Systems Biology and Intelligent Computing*

[7] Banfield, E.R., Student Member, IEEE,Lawrence O. Hall, Fellow, IEEE,Kevin W. Bowyer, Fellow, IEEE, and W.P. Kegelmeyer, Member, IEEE, JANUARY 2007. A Comparison of Decision Tree Ensemble Creation Techniques, *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, VOL. 29, NO. 1.

[8] Sundaram, Shanmuga, M., Anand Kumar M, and Soman, K.P. 2015. AMRITA CEN@ SemEval-2015: Paraphrase Detection for Twitter using Unsupervised Feature Learning with Recursive Autoencoders. *SemEval*-2015 (2015): 45.

[9] Mahalakshmi, S., Anand Kumar, M., Soman, K.P. 2015. Paraphrase detection for Tamil language using deep learning algorithm, (2015) *International Journal of Applied Engineering Research*, 10 (17), pp. 13929-13934

[10] Socher, R., Huang, E. H., Pennin, J., Manning, C.D., and Andrew, Y. Ng. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. *Advances in Neural Information Processing Systems* (pp. 801-809).

[11] Anand Kumar, M., Singh, S., Kavirajan, B., and Soman, K. P. 2016. DPIL@FIRE2016: Overview of shared task on Detecting Paraphrases in Indian Languages, Working notes of FIRE 2016 - *Forum for Information Retrieval Evaluation*, Kolkata, India, December 7-10, CEUR Workshop Proceedings, CEUR-WS.org.