HIT2016@DPIL-FIRE2016:Detecting Paraphrases in Indian Languages based on Gradient Tree Boosting

Leilei Kong*
[1]College of Information and Communication Engineering, Harbin Engineering University, Harbin, China
[2]School of Computer Science and Technology, Heilongjiang Institute of Technology, Harbin, China;
+86 451 88028910
kongleilei1979@gmail.com

Kaisheng Chen
School of Computer Science and Technology, Heilongjiang Institute of Technology, Harbin, China;
+86 451 88028910
kaishengchen1997@outlook.com

Liuyang Tian
College of Information and Communication Engineering, Harbin Engineering University, Harbin, China
+86 451 88028910
tianliuyang2016@outlook.com

ZhenyuanHao
School of Computer Science and Technology, Heilongjiang Institute of Technology, Harbin, China;
+86 451 88028910
zhenyuan_hao@163.com

Zhongyuan Han
School of Computer Science and Technology, Heilongjiang Institute of Technology, Harbin, China;
+86 451 88028910
Hanzhongyuan@gmail.com

Haoliang Qi
School of Computer Science and Technology, Heilongjiang Institute of Technology, Harbin, China;
+86 451 88028910
haoliang.qi@gmail.com

ABSTRACT

Detecting paraphrase is an important and challenging task. It can be used in paraphrases generation and extraction, machine translation, question and answer and plagiarism detection. Since the same meaning of a sentence is expressed in another sentence using different words, it makes the traditional methods based on lexical similarity ineffective. In this paper, we describe a strategy of Detecting Paraphrases in Indian Languages, which is a workshop track proposed by Forum Information Retrieval Evaluation 2016. We formalize this task as a classification problem, and a supervised learning method based on Gradient Boosting Tree is utilized to classify the types of paraphrase plagiarism. Inspired by the Meteor evaluation metrics of machine translation, the Meteor-like features are used for the classifier. Evaluation shows the performance of our approach, which achieved the highest Overall Score (0.77), the highest F1 measure for both Task1 and Task2 on Malayalam and Tamil, and the highest F1 measure on Punjabi Task2 in the 2016 FIRE Detecting Paraphrase in Indian Languages task.

CCS Concepts

• **Information systems→Information retrieval**

Keywords

Paraphrase; Classification; Indian Languages; Gradient Tree Boosting.

1. INTRODUCTION

Detecting Paraphrasing has attracted the attention of researchers in recent years. It is widely used in paraphrases generation and extraction, machine translation, question and answer and plagiarism detection.

In the task description of *Detecting Paraphrases in Indian Languages* of Forum Information Retrieval Evaluation 2016 (FIRE 2016)[1], the paraphrase is defined as "the same meaning of a sentence is expressed in another sentence using different words". The proposed task is focused on sentence level paraphrase identification for Indian languages (Tamil, Malayalam, Hindi and Punjabi). There are two tasks are proposed by FIRE. The first sub task is: given a pair of sentences from newspaper domain, the task is to classify them as paraphrases (P) or not paraphrases (NP), and the second one is: given two sentences from newspaper domain, the task is to identify whether they are completely equivalent (E) or roughly equivalent (RE)[1] or not equivalent (NE) [6].

The paraphrased sentences always retain the semantic meaning and usually obfuscated by manipulating the text and changing most of its appearance. The words in the original sentence is replaced with synonyms/antonyms, and short phrases are inserted to change the appearance, but not the idea, of the text (Alzahrani et al., 2012). Otherwise, the sentence reduction, combination, restructuring, paraphrasing, concept generalization, and concept specification also are used to paraphrase the sentence. All of these operations make the paraphrases identification difficult, because it involves the semantic similarity, lexical comprehension, syntactical identification, morphological analysis, and so on.

Since the appearance have changed beyond recognition in paraphrased sentence, the methods only relying on the term matching or single feature may be become ineffective in detecting paraphrase. More features should be integrated in the model to detecting paraphrase. So we consider a machine learning method based on classification to address this problem.

Intuitively, the former sub tasks can be viewed as a two-category classification and the latter is multi-category classification. If we formalize the task of detecting paraphrase as a classification problem, our objectives focus on answeringthe following two questions: (1) Which classification-based methods can effectively be applied to the detecting paraphraseproblem, and (2) which features should be used in the classifier.

For the first problem, we choose Gradient Tree Boosting to learn t he classifier [2,3]. Regarding the second issues, inspired by the METEOR evaluation metrics of machine translation [4], we design

[1]http://nlp.amrita.edu/dpil_cen/

* Corresponding author

the METEOR-like features for our classifier. Integrating some classical similarity measure feature, we develop the feature set.

Using the training and testing corpora of Detecting Paraphrases in Indian Languages proposed by FIRE, we rigorously evaluate various aspects of our classification method for detecting paraphrases. Experimental results show that the proposed method can effectively classify the paraphrases pairs.

The rest of this paper is organized as follows. In Section 2, we ana lyze the problem of Detecting Paraphrases in Indian Languages, in troduce the model we used, and describe the features which the cl assifier uses. In Section 3, we report the experimental results and performance comparisons with the other detection methods. And i n the last section we conclude our study.

2. CLASSIFICATION FOR DPIL

We now explore machine-learning methods for Detecting Paraphrases in Indian Languages. In this section, we analyze the main issues of Detecting Paraphrases in Indian Languages firstly. And then a classification method based on boosting tree is proposed. Finally, we describe the features which the classifier used.

2.1 Problem Analysis

As we have discussed in above section, paraphrases identification is difficult to detect. The traditional similarity computing methods, such as Cosine Distance, Jaccard Coefficient, Dice Distance, may be ineffective for paraphrases. Figure 1 exemplifies the paraphrases cases.

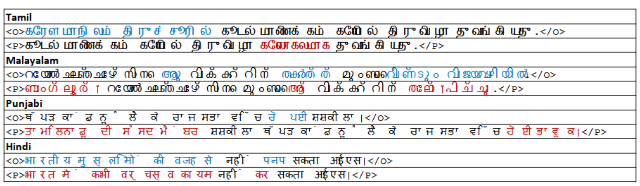


Figure 1.A paraphrases cases

From Figure 1, we can see that the two sentences having the paraphrasing relationship are different in their appearance. Furthermore, we conduct the analysis on 1000 randomly selected cases with paraphrase relationship on Malayalam sub corpora and all four languages corpora. Figure 2 displays the distribution with Jaccard Coefficient and METEOR-F1 as y-coordinate.

It is easy to detect from Figure 2 that the scores of Jaccard coefficient are all very low, the average score is only 0.1332. Since there are few the same terms between the two sentence, only considering the term similarity may be inadequate. We analysis for identifying the relationship of them, more feature should be considered.

2.2 Problem Definition

According the description of detection paraphrases, we formalize the problem as follows. Denote a pair sentences as $s_i=(o_i, p_i)$, where o_i is the original sentence and p_i is the paraphrased sentence. Note that given a pair (o_i, p_i) on the training data, we can get its label, which make learn a model for classification possible. Let the train corpora $D=\{(x_1,y_1), (x_2,y_2), \ldots, (x_i,y_i), \ldots, (x_n,y_n)\}$, where $x_i \in \mathbb{R}^N$ is a feature vector of s_i and

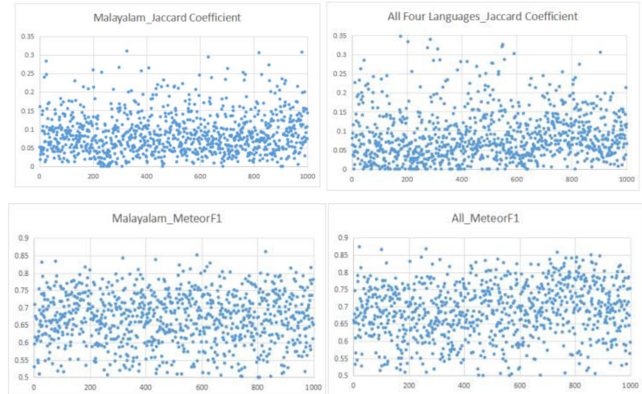


Figure 2. Score distribution of Jaccard coefficient on Malayalam (up) sub corpora and all four languages corpora(down)

$x_i=(x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)})^T, i=1,2,\ldots,N$. We use a function to get each x_i defined as follows.

$$x_{(i)}=\Phi(o_i, p_i) \qquad (1)$$

where $x_{(i)}=\Phi(o_i, p_i)$ is a mapping onto features that describes the paraphrase between the i-th original sentence o_i and the paraphrased sentence p_i.

And y_i is the label of x_i to denote the category of each x_i. For the task 1, we define $y_i \in \{P, NP\}$, and for task 2, we define $y_i \in \{E, RE, NE\}$.

Then the framework of learning problem can be depicted in Figure 3.

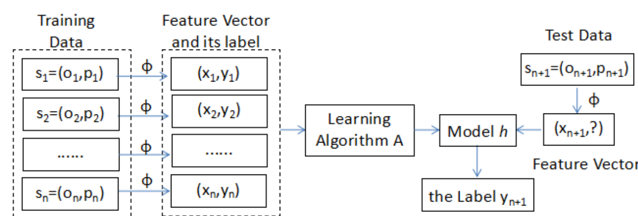


Figure 3. The framework of Detection Paraphrase

Then, given D as training data, the learning system will learn a condition probability P(Y|X) based on the training data. Then given a new input x_{n+1}, the classification system gives the corresponding output label y_{n+1}according to the learned classifier.

2.3 Classification Model: Gradient TreeBoosting

Boosting tree is one of the best methods to improve the performan ce of statistical learning [2,3]. In this experiment, we use the Gradient Tree Boosting as the classification algorithm to learn the classifier. Gradient boosting is typically used with decision trees (especially CART trees) of a fi xed size as base learners.

2.4 Features

There are two groups of features, the similarity-based features and the METEOR-like features, are utilized to define $x_{(i)}=\Phi(o_i, p_i)$.

The similarity-based features are used to capture the matching degree of o_i and p_i, and METEOR-like features is used to describe the semantic similarity. Specially, the METEOR-like features is inspired by METEOR, the measure metrics for machine translation, which is used to evaluate the performance of a translator. Table 1 list these features in detail.

Table 1. Features for detecting paraphrases

Features	Computing methods	Description
Jaccard Coefficient	$JC(s_i, r_j) = \frac{\lvert s_i \cap r_j \rvert}{\lvert s_i \cup r_j \rvert}$	The ratio of number of shared terms against total number of terms.
Cosine Similarity	$CS(\vec{x}_i, \vec{y}_i) = \frac{\vec{x}_i \cdot \vec{y}_i}{\lVert \vec{x}_i \rVert \cdot \lVert \vec{y}_i \rVert}$	$\vec{x}_i \cdot \vec{y}_i$ is the inner product of x and y, and $\lVert \vec{x} \rVert$ represents the length of vector.
Dice Coefficient	$DC(s,r) = \frac{2 \cdot common(s,r)}{len(s) + len(r)}$	common (s, r) is the total number of the common unigrams in s and r, and len(r) and len(s) are the total number of unigrams in r and s.
METEOR Precision	$P = \frac{common(s,r)}{len(r)}$	common (s, r) is the total number of the common unigrams in s and r, and len(r) is the total number of unigrams in r.
METEOR Recall	$R = \frac{common(s,r)}{len(s)}$	len(s) is the total number of unigrams in s.
METEOR F1	$F1 = \frac{2PR}{R+P}$	Combine the precision and recall.
METEOR Fmean	$Fmean = \frac{10PR}{R+9P}$	Combine the precision and recall.
METEOR Penalty	$Penalty = 0.5 * \left(\frac{len(chunks)}{common(s,r)}\right)^3$	len(chunks)is the number of the longer matchesin each chunk.
METEOR score	$Score = Fmean * (1 - Penalty)$	The overall METEOR score.

3. Experiments

3.1 Dataset

The evaluation dataset is the Detecting Paraphrase in India Language (DPIL) which is mainly obtained from the newspaper. The details of this corpora can be found in http://nlp.amrita.edu/dpil_cen/.

The corpora are divided into two different subsets: Task1-set and Task2-set, and each sub set contains four different categories India language: Tamil, Malayalam, Hindi and Punjabi. The Task1-set contains 12400 samples, including 9200 training samples and 3200 test samples, and the Task2-set contains 17650 examples, including 12700 training samples and 4950 test samples. The statistics of training and testing data is shown in Table 2 and Table 3.

Table 2. Corpus statistics of DPIL 2016 on Task1

		Train					Test				
Language		Hin	Mal	Pun	Tam	all	Hin	Mal	Pun	Tam	all
SampleNumber		2500	2500	1700	2500	9200	900	900	500	900	3200
Avg terms	blank	32	18	39	24	27	32	19	43	23	28
	4gram	126	166	150	175	155	120	181	164	176	160

Table 3. Corpus statistics of DPIL 2016 on Task2

		Train					Test				
Language		Hin	Mal	Pun	Tam	All	Hin	Mal	Pun	Tam	all
SampleNumber		3500	3500	2200	3500	12700	1400	1400	750	1400	4950
Avg terms	blank	34	18	41	24	28	42	19	41	28	31
	4gram	131	164	156	178	158	154	177	157	207	176

3.2 Experimental Settings

3.2.1 Pre-processing

For each sentence pair in training data and test data, wefirstly remove numbers, punctuation and blank spaces. Then, we adopt two types of word segmentation, one is taking each word as a term unit, and the other is based on the n-gram, which the words in sentence are segmented in the form of n-gram. For example, Figure 4 shows an example of 4-gram. In the experiments, the n is set empirically.

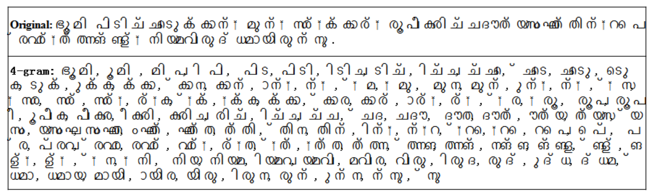


Figure 4. The example of 4-gram

3.2.2 Parameter Tuning

On the training corpus, the classifier is trained by using sklearn Boosting Classifier Gradient[2]. The learning rate (learning rate shrinks the contribution of each tree by learning rate) is set as 1.0, the max_depth (the maximum depth limits the number of nodes in the tree) is set as 1, the random state (random state is the seed used by the random number generator) is set as 0. All the other parameters are set as their default values except the parameter n_estimators (The number of boosting stages to perform).

The other parameters, including the methods of word segmentation, the method of pre-processing method, the n value of ngram, are set experimentally.

We use the cross validation to tune the parameter n_estimators. The training corpora is randomly divided into two equal parts, and one is chosen as the training data and the other as the validation data.

3.3 Performance Measures

In this evaluation experiment, the experimental results are evaluated according to [5].

1) TP: The sample is true, and the results obtained are positive.
2) FP: The sample is false, and the results obtained are positive.
3) FN: The sample is false, and the results obtained are negative.
4) TN: The sample is true, and the results obtained are negative.

According to the above measure metrics, the Precision and Recall are defined as follows:

$$\text{precision} = \frac{TP}{TP + FP} \quad (5)$$

$$\text{recall} = \frac{TP}{TP + FN} \quad (6)$$

The main evaluation metrics adopted by DPIL is Accuracy and F1 measure defined as follows:

[2]http://scikit-learn.org/stable/

$$\text{accuracy} = \frac{TP + TN}{TP + FN + FP + TN} \quad (7)$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall} \quad (8)$$

3.4 Experimental Results

3.4.1 Experimental results on sub corpora

Table 4 show the experimental results released by FIRE.

Table 4. Experimental results on DPIL@FIRE2016

(a) Task 1 sub corpus

TEAM	Accuracy				F1 Measure			
	Mal	Tam	Hin	Pun	Mal	Tam	Hin	Pun
HIT2016	0.8377	0.8211	0.8966	0.9440	0.8100	0.7900	0.8900	0.9400
KS_JU	0.8100	0.7888	0.9066	0.9460	0.7900	0.7500	0.9000	0.9500
NLP-NITMZ	0.8344	0.8333	0.9155	0.9420	0.7900	0.7900	0.9100	0.9400
JU-NLP	0.5900	0.5755	0.8222	0.9420	0.1600	0.0900	0.7400	0.9400
Anuj	——	——	0.9200	——	——	——	0.9100	——
DAVPBI	——	——	——	0.9380	——	——	——	0.9400
BITS-PILANI	——	——	0.8977	——	——	——	0.8900	——
NLP@KEC	——	0.8233	——	——	——	0.7900	——	——
ASE	——	——	0.3588	——	——	——	0.3400	——
CUSAT TEAM	0.8044	——	——	——	0.7600	——	——	——
CUSAT NLP	0.7622	——	——	——	0.7500	——	——	——

(b) Task 2 sub corpus

TEAM	Accuracy				F1 Measure			
	Mal	Tam	Hin	Pun	Mal	Tam	Hin	Pun
HIT2016	0.7486	0.7550	0.9000	0.9226	0.7460	0.7398	0.8984	0.9230
KS_JU	0.6614	0.6735	0.8521	0.8960	0.6578	0.6645	0.8482	0.8960
NLP-NITMZ	0.6243	0.6571	0.7857	0.8120	0.6068	0.6307	0.7642	0.8086
JU-NLP	0.4221	0.5507	0.6857	0.8866	0.3078	0.4319	0.6841	0.8866
Anuj	——	——	0.9014	——	——	——	0.9000	——
DAVPBI	——	——	——	0.7466	——	——	——	0.7274
BITS-PILANI	——	——	0.7171	——	——	——	0.7123	——
NLP@KEC	——	0.6857	——	——	——	0.6674	——	——
ASE	——	——	0.3543	——	——	——	0.3535	——
CUSAT TEAM	0.5086	——	——	——	0.4658	——	——	——
CUSAT NLP	0.5207	——	——	——	0.5130	——	——	——

The experimental results show that the proposed method achieves the best Accuracy on Malayalam of Task 1 and on Malayalam, Tamil and Punjabi of Task 2. And the highest F1 measure for both Task1 and Task2 on Malayalam and Tamil, and the highest F1 measure on Punjabi Task2 in the 2016FIREDetecting Paraphrase in Indian Languages task.

3.4.2 Effect of word segmentation

For the word segmentation, we utilize two processing methods. One is based on the space to do the word segmentation, and the other is based on n-gram. We compare the two kinds of word segmentation methods in Table 5.

Table 5. Comparison of two different preprocessing

Task1	4-gram				space			
	Mal	**Tam**	**Hindi**	**Pun**	**Mal**	**Tam**	**Hindi**	**Pun**
Precision	0.8993	0.9587	0.9235	0.9884	0.8771	0.9543	0.9340	0.9911
Recall	0.9301	0.9606	0.9187	0.9921	0.9279	0.9574	0.9289	0.9921
Accuracy	0.8957	0.9517	0.9054	0.9885	0.8785	0.9469	0.9178	0.9901
F1	0.9143	0.9596	0.9210	0.9902	0.9017	0.9558	0.9314	0.9916
Task2	**4-gram**				**space**			
	Mal	**Tam**	**Hindi**	**Pun**	**Mal**	**Tam**	**Hindi**	**Pun**
Precision	0.7298	0.7873	0.8499	0.9810	0.7135	0.7917	0.8553	0.9814
Recall	0.7370	0.7918	0.8484	0.9808	0.7227	0.7949	0.8545	0.9813
Accuracy	0.7370	0.7918	0.8484	0.9808	0.7227	0.7949	0.8545	0.9813
F1	0.7309	0.7878	0.8483	0.9808	0.7134	0.7923	0.8541	0.9813

From the experimental results, we can see that the method of 4-gram segmentation achieves higher F1 Measure than the space segmentation, so we use n-gram method in the following experiments to deal with the India corpus.

3.4.3 Effects of pre-processing

In our experiment, there are two types of pre-processing methods. To investigate the different contribution of each pre-processing method on each language, we analyze the effects of pre-processing. Taking 4gram word segmentation as example, Table 6 gives the experimental results, where *removing all* means remove the punctuation, the number and the space, and *reserving* * means reserving * and removing all others. For example, *reserving punctuation*represents the punctuation is reserved and the number and space are removed.

Table 6. Effects of pre-processing

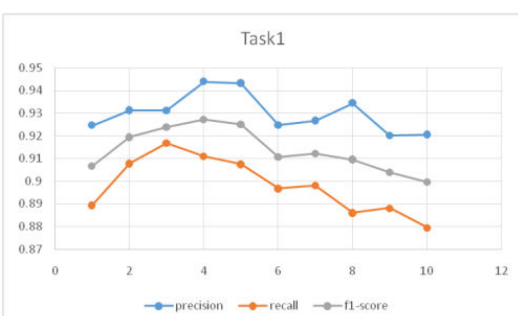
Mal		**Reserved punctuation**	**Reserved number**	**Reserved space**	**Remove all**
Task1	Precision	0.9013	0.8995	0.8992	0.8988
	Recall	0.9280	0.9276	0.9325	0.9335
	Accuracy	0.8956	0.8944	0.8966	0.8968
	F1 Measure	0.9144	0.9133	0.9154	0.9157
Task2	Precision	0.7304	0.7258	0.7253	0.7289

	Recall	0.7380	0.7340	0.7321	0.7362
	Accuracy	0.7380	0.7340	0.7321	0.7362
	F1 Measure	0.7316	0.7273	0.7264	0.7299
Tam		**Reserved punctuation**	**Reserved number**	**Reserved space**	**Remove all**
Task1	Precision	0.9585	0.9591	0.9535	0.9570
	Recall	0.9593	0.9590	0.9558	0.9607
	Accuracy	0.9506	0.9507	0.9455	0.9506
	F1 Measure	0.9589	0.9590	0.9546	0.9588
Task2	Precision	0.7855	0.7874	0.7864	0.7871
	Recall	0.7901	0.7915	0.7897	0.7917
	Accuracy	0.7901	0.7915	0.7897	0.7917
	F1 Measure	0.7861	0.7880	0.7866	0.7880
Hindi		**Reserved punctuation**	**Reserved number**	**Reserved space**	**Remove all**
Task1	Precision	0.9218	0.9242	0.9310	0.9230
	Recall	0.9136	0.9151	0.9244	0.9195
	Accuracy	0.9018	0.9039	0.9133	0.9054
	F1 Measure	0.9176	0.9195	0.9275	0.9211
Task2	Precision	0.8490	0.8502	0.8495	0.8500
	Recall	0.8477	0.8481	0.8487	0.8486
	Accuracy	0.8477	0.8481	0.8487	0.8486
	F1 Measure	0.8475	0.8480	0.8484	0.8484
Pun		**Reserved punctuation**	**Reserved number**	**Reserved space**	**Remove all**
Task1	Precision	0.9909	0.9904	0.9867	0.9903
	Recall	0.9914	0.9908	0.9895	0.9905
	Accuracy	0.9895	0.9889	0.9859	0.9887
	F1 Measure	0.9911	0.9906	0.9881	0.9904
Task2	Precision	0.9810	0.9774	0.9812	0.9812
	Recall	0.9808	0.9772	0.9810	0.9811
	Accuracy	0.9808	0.9772	0.9810	0.9811
	F1 Measure	0.9808	0.9772	0.9810	0.9811

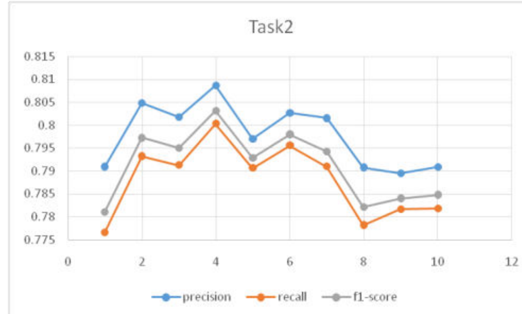
According to the experimental results shown in Table 6, even thoughwe find that there are few differences when we removing punctuation, numbers and spaces, we still accept the best pre-processing method on the test dataset.

3.4.4 *Effects of n-gram*

For analyze the effects of n, we carry out the experiments from 1-gram to 10-gram, and with Precision, Recall and F1 measure as evaluation indicators. The experimental results are shown in Figure 5.



(a) The experimental results on Task 1



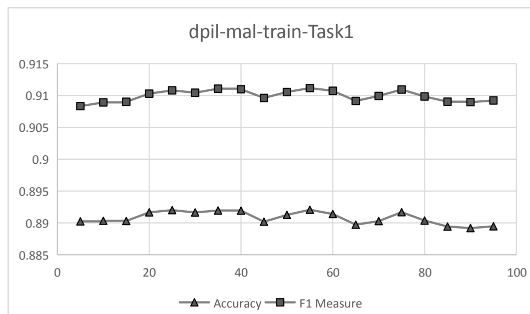
(b) The experimental results on Task 2

Figure 5. The effects of n-gram

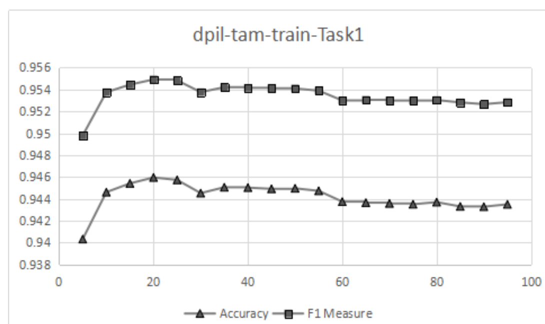
According to the above experimental results, 4-gram achieves the best results. So we set n=4 in the testing corpora of DPIL 2016.

3.4.5 *Effects of n_estimators*

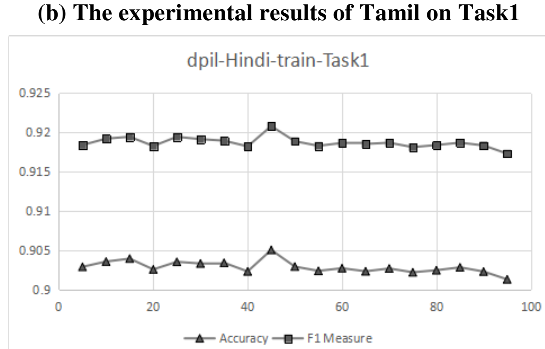
The parameter n_estimators is the number of iterations of boosting stage when the classification model trained. It is set empirically. Figure 6 shows the results on training datasets.



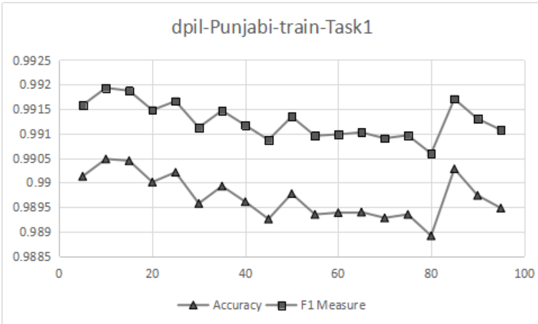
(a) The experimental results of Malayalam on Task1



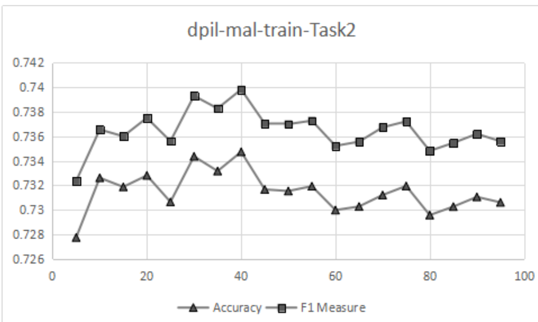
(b) The experimental results of Tamil on Task1



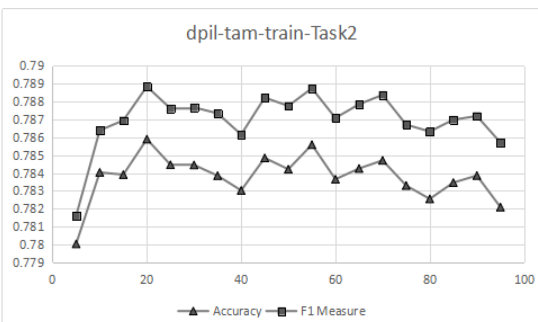
(c) The experimental results of Hindion Task1



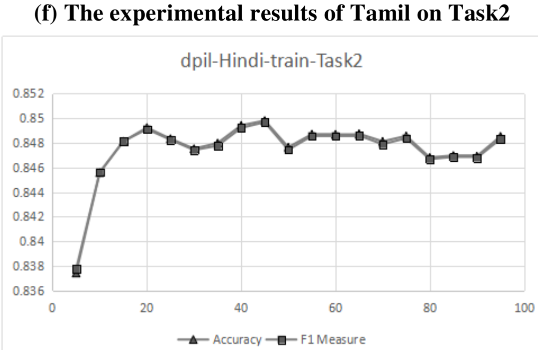
(d) The experimental results of Punjabi on Task1



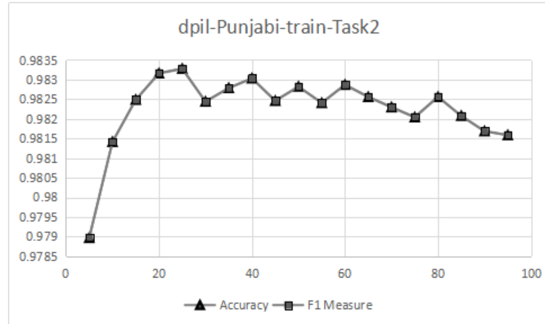
(e) The experimental results of Malayalam on Task2



(f) The experimental results of Tamil on Task2



(g) The experimental results of Hindion Task2



(h) The experimental results of Punjabi on Task2

Figure 6.Effects of n_estimators

According to Figure 6, we get the value of the parameter n_estimators of each language. Details are shown in Table 7 which is used in the testing datasets of DPIL.

Table 7.N_estimatorssetting

	Task1	Task2
Malayalam	55	40
Tamil	20	20
Hindi	45	45
Punjabi	10	25

4. CONCLUSIONS

We describe an approach to the Detecting Paraphrase problem in India Language that makes used of the Gradient Tree Boosting. Overall, the approach was very competitive and achieved the highest Accuracy and F1 measure among all task participants.

5. ACKNOWLEDGMENTS

This work is supported by Youth National Social Science Fund of China (No. 14CTQ032), National Natural Science Foundation of China (No. 61370170), and Research Project of HeilongjiangProv incial Department of Education (No. 12541677, 12541649).

6. REFERENCES

[1] Alzahrani, S. M., Salim, N., and Abraham, A. 2012. Understanding plagiarism linguistic patterns, textual features, and detection methods. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42(2), 133-149.

[2] Friedman, J. H., 2001. Greedy function approximation: a gradient boosting machine. Annals of statistics, 1189-1232.

[3] Friedman, J. H., 2002. Stochastic gradient boosting. Computational Statistics & Data Analysis, 38(4), 367-378.

[4] Banerjee, S., andLavie, A., 2005, June. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization. 29: 65-72.

[5] Li, Hang., 2012. Statistical learning methods.Tsinghua university press(in Chinese).

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