Modeling Classifier for Code Mixed Cross Script Questions

Rupal Bhargava[1] Shubham Khandelwal[2] Akshit Bhatia[3]

Yashvardhan Sharma[4]

WiSoc Lab, Department of Computer Science
Birla Institute of Technology and Science, Pilani Campus
Pilani-333031

{rupal.bhargava[1], f2013131[2], f2013722[3],yash[4]} @pilani.bits-pilani.ac.in

ABSTRACT

With a boom in the internet, the social media text had been increasing day by day and the user generated content (such as tweets and blogs) in Indian languages are written using Roman script due to various socio-cultural and technological reasons. A majority of these posts are multilingual in nature and many involve code mixing where lexical items and grammatical features from two languages appear in one sentence. Focusing on this current multilingual scenario, code-mixed cross-script (i.e., non-native script) data gives rise to a new problem and presents serious challenges to automatic Question Answering (QA) and for this question classification will be required which is an important step towards QA. This paper proposes an approach to handle cross script question classification as it is an important task of question analysis which detects the category of the question.

CCS Concepts

•**Information systems** → *Information integration; Data analytics; Data mining; Social recommendation; Query representation; Query intent;*

Keywords

Code Mixing, Code Switching, Question Classification, Machine Learning

1. INTRODUCTION

With the proliferation of social network large volumes of text is being generated daily. Traditional machine learning algorithms used for text analysis such as Named Entity Recognition (NER) or POS Tagging or parsing, are language dependent.These algorithms usually achieve their objective using co-occurrence patterns of features. Due to such language dependence, it has been observed by many studies that a variety of problems related to social media text are hindered. One such problem is Question Answering (QA). Being a classic application of NLP, Question Answering (QA) has practical applications in various domains such as education, health care, personal assistance etc. QA is a retrieval task which is more challenging than the task of common search engine because the purpose of QA is to find accurate and concise answer to a question rather than just retrieving relevant documents containing the answer [6]. Recently, Banerjee et al. [2] formally introduced the code-mixed cross-script QA problem. The first step of understanding a question is to perform a question analysis. Question classification is an important task of question analysis which detects the answer to the type of the question. Question classification helps not only filter out a wide range of candidate answers but also determine answer selection strategies [6]. Furthermore, it has been observed that the performance of question classification has significant influence on the overall performance of a QA system.

The Subtask 1 in the shared task on Mixed Script Information Retrieval in FIRE-2016 addresses the task of code mixed cross script question classification where 'Q' represents set of factoid questions written in Romanized Bengali along with English. The task is to classify each given question into one of the predefined coarse-grained classes. This paper proposes an algorithm for solving question classification task proposed by MSIR, FIRE 2016 Subtask 1 organizers, using different machine learning algorithms.

Rest of the paper is organized as follows. Section 2 explains the related work that has been done in the past few years. Section 3 presents the analysis of dataset provided by MSIR 2016 Task Organizers [1]. Section 4 explains the methodology that have been performed for the task with flowcharts to explain the flow. Section 5 describes the algorithm proposed for question classification. Section 6 elaborates the evaluation and experimental results and error analysis. Section 7 concludes the paper and presents future work.

2. RELATED WORK

Today social media platforms are flooded by millions of posts everyday on various topics resulting in code mixing in multilingual countries like India. A lot of work had been done in FIRE 2015 for language identification in cross script information retrieval. Bhattu et al. [4] proposed a two stage algorithm, where in the first stage sentence level n-grams based classifiers and in the second stage word level n-grams classifiers were used. Bhargava et al. [3] proposed a hybrid approach to do query labeling by generating char n-grams as features and using logistic regression for language labeling. For question analysis of such data, Question classification is done to understand the question that allows determining some constraints the question imposes on a possible answer. Zhang et al. [8] used bag of words and bag of n-grams as features and applied K-NN, SVM, Naive Bayes to automate question classification and concluded that with surface text features the SVM outperforms the other classifiers. Banerjee et al. [2] proposed a QA system which takes cross-script (non-native) code-mixed questions and provides a list of information response to automate the question answering. Corpus acquisition was done from social media, question acquisition using a cloud based service without getting bias, corpus annotations and an evaluation scheme suitable to the corpus annotation. Li et al. [6] proposed question classification using the role of semantic information developing a hierarchical classifier guided by a layered semantic hierarchy of answer types.

3. DATA ANALYSIS

The training data provided [1], consisted of 330 questions labeled with its specific coarse-grained question type classes. In total there are 9 different question type classes in the data set. As shown in Figure 1, class-types 'ORG' and 'TEMP',

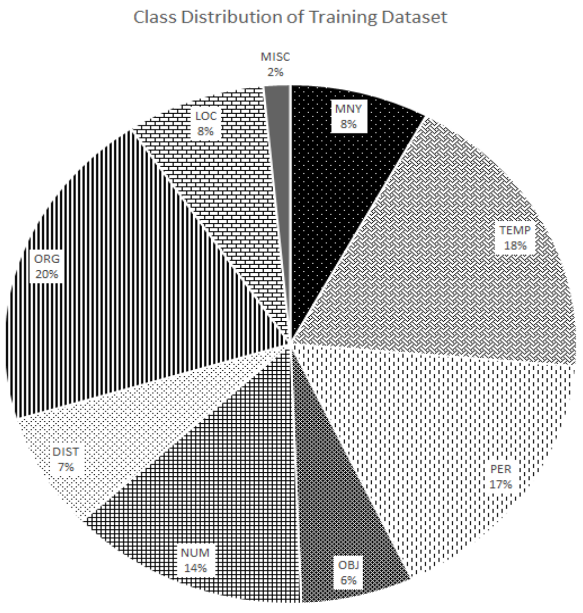


Figure 1: Class Distribution of Training Data Set

comprises majority of the instances. Each of these classes represents a particular type of question related to specific entities. Class type 'MNY' stands for Money related questions and the instances comprises of words like 'fare', 'price' and helping words like 'koto' (bn) and how much etc. Class type 'PER' stands for Person related questions mostly comprising of words like 'who', 'whom' etc. implying for the subject of the sentence being a person. Class type 'TEMP' implies time related questions mainly comprising of words like 'when', 'at' etc. Class type 'OBJ' stands for the Entity/Object implying that subject of the sentence is an entity and mainly comprising of words like 'what', 'kon' etc. Class type 'NUM' stands for Numeric entity related questions and mainly involves usage of words like 'how many', 'koto' etc. Class type 'DIST' stands for Distance and implies that question is related to distance between places. Class type 'LOC' stands for Location and thereby mainly comprises of words like 'where', 'jabe' etc. Class type 'ORG' stands for Organization and relates to questions centered on particular organization, team or any other group of people and these questions mainly comprises of words like 'which', 'what', 'team' etc. Class type 'MISC' stands for Miscellaneous; this class has the minimum representation in the data set and relates to a variety of questions.

The entire data set has sentences in a code-mixed format, consisting of words which either belong to Bengali or English language. The data set does not contain any code-mixing done at word level. Also there are no punctuation in the data set except the question mark (?) while there are a lot of named entities (belonging to both English and Bengali) present in it.

4. PROPOSED TECHNIQUE

A word-level n-gram based approach is used to classify code mixed cross script question records (comprising of words belonging to both English and Bengali languages), into nine different coarse grained question type classes. The proposed methodology involves a pipelined deployment of different techniques as mentioned in Figure 2. Proposed technique can be majorly divided into the following four phases:

1. Pre-processing
2. Named-entity recognition and removal
3. Translation
4. Classification

4.1 Pre-processing

Data is pre-processed for label separation and case conversion for the efficient application of the classifiers. The pre-processing techniques were deployed as follows:

1. *Separation of Class labels and Training Set Entries*: The data set comprised of mixed script question records labeled with specific class type. These question entries and the respective class labels were segregated, on the basis of position of question mark symbol in the data entries. This segregation was done so that separate feature vectors of question records and class labels could be formed, as per the deployment requirements of the classifiers.
2. *Case Conversion*: For the purpose of normalization, all the data entries were converted into lower case. This technique involved identification and replacement of the upper case letters with their lower case counterparts by means of manipulation of the ASCII code values.

4.2 Named-entity recognition and removal

The pre-processed data set comprised of entries which had a large number of named-entities. Named-entities in a text can be referred to pre-defined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages etc. In English language named entities occur in certain manner at certain positions according to sentence structure. But when it comes to multi-lingual sentences, sentence structure varies a lot. Named Entities are identified using a dictionary based approach. The data set used for NER mainly comprised of the entries from FIRE 2015 Subtask1's data set [5]. This data set contained entries belonging to both Bengali and

English languages. For the purpose of classification of the question records into one of the class types, the presence of these named-entities was irrelevant, as these entities did not contribute in building question structure for class-type determination, and hence their removal was mandatory.

4.3 Translation

After the initial two phases, the remaining Bengali words were transliterated into their native scripts and then further translated to their respective English counterparts using the Google translation API [1]. This technique helped to create a monolingual, single-script data set from the mixed script data set provided so that the efficient application of classifiers could take place. Using this approach, different code mixed cross script variants (each variant using different combination of words belonging to either Bengali or English languages) of the same question record were translated and hence standardized to only one question record (in English language). For example the question record "Hazarduari te koto dorja ache?" and the record "Hazarduari te how many dorja ache?", both refer to the same question but use different combination of words, and hence standardizing this to the English translation, would lead to an increase in the accuracy.

4.4 Classification

The proposed approach uses the data set obtained from translation phase and deploys the technique of n-grams to form the feature vectors for each record in the data set. The approach follows a word-level implementation of *n-grams* with 'n' being varied in the range 2 to 4, and thereby generation of feature vectors for each question record in the training set. The transposed matrix of these feature vectors along with the numerically encoded class label matrix is then used as inputs to classifiers [7]. For the three runs, the following different classifiers are used:

1. Gaussian Naive Bayes Classifiers
2. Logistic Regression Classifier
3. Random Forest Classifier with Random State = 1

5. ALGORITHM

Algorithm 1 explains the proposed technique. Data set comprising of mixed script question records along with their respective class labels, is used as input. This input is pre-processed by deploying the techniques for separation of class labels from data entries (implemented by the function: Label_Separation()) and case conversion of data records into lower cases (using function: Case_Conversion()). Named entities (NE) are removed from this pre-processed data (implemented by the function: NE_Removal()). The remaining Bengali words present in the data set are then translated to their respective English equivalents by using Google Translation API [1] (by means of the function: Translation()). The technique of n-grams is then applied on this data set to form the corresponding feature vectors. First a vector for converting the textual entries into a matrix of n-gram token counts (word level n-grams with n in the range 2 to 4) is created (implemented by the function Count_Vectorizer()). This vector is then used to generate another callable (by function: Build_Analyzer()) which is used to produce n-grams tokens when called on each data set rows (implemented by callable: Analyser()) and word level n-grams corresponding to each row is appended (by means of function append()) to the n-gram list. Feature vectors for the data entries are generated using this n-gram list (implemented by the function: Create_Feature_Vector()). The class labels for the training purpose are numerically encoded (by means of function: Encode_Class()) and then corresponding feature vectors for these class labels are generated. These two sets of feature vectors are then used as inputs to the classifier (implemented by the function: Classifier()). Classifier() function can be replaced by functions of different classifier like GaussianNB, LogisticRegression or RandomForestClassifier. The classifier is then used to predict the class labels generated as output.

[1]https://translate.google.com/

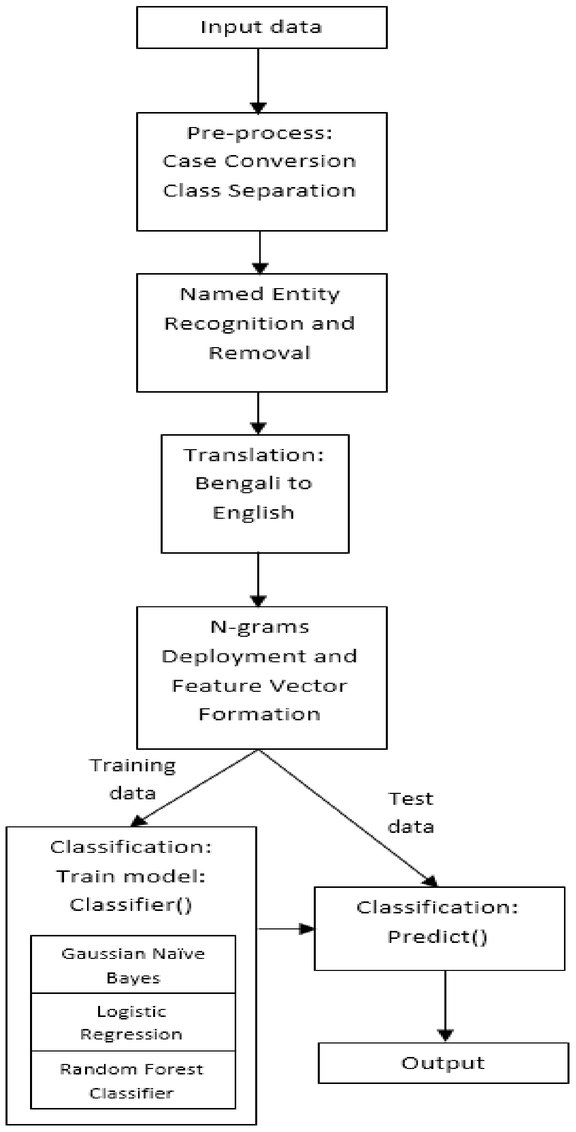

Input data
Pre-process: Case Conversion Class Separation
Named Entity Recognition and Removal
Translation: Bengali to English
N-grams Deployment and Feature Vector Formation
Training data
Test data
Classification: Train model: Classifier()
Gaussian Naïve Bayes
Logistic Regression
Random Forest Classifier
Classification: Predict()
Output

Figure 2: Block Diagram for Proposed Technique

Algorithm 1 Algorithm for Detecting paraphrases

```
1: Input: Mixed Script (bn+en) Question records, S, Training class labels, T
2: Output: Predicted class labels, P
3: Initialization: P=[], n_grams=[]
4: for i=0 to S.length do
5:     Label_Separation(S[i])
6:     Case_Conversion(S[i])
7:     NE_Removal(S[i])
8:     Translation(S)
9: end for
10: Vectorizer = Count_Vectorizer(ngram_range=(2,4))
11: Analyzer = Vectorizer.build_analyzer()
12: for i=0 to S.length do
13:     row=Analyzer(S[i])
14:     for j=0 to row.length do
15:         n_grams.append(row[j])
16:     end for
17: end for
18: Matrix_Data=Create_Feature_Vector(n_grams)
19: Class_List=Encode_Class(T)
20: Matrix_Class= Create_Feature_Vector(Class_List)
21: clf =Classifier(Matrix_Data, Matrix_Class)
22: clf.fit(Matrix_Data, Matrix_Class)
23: P=clf.predict(Matrix_Test)
```

6. EXPERIMENTS

MSIR, FIRE 2016, Subtask 1 involved classification of mixed-script (Bengali and English) questions into nine different coarse grained question type classes as discussed in Section 3. The training dataset comprised of 330 records (along with class labels) and it was used to classify a test dataset comprising of 180 mixed script question records. Total seven teams from different institutes of the country participated in the process and each team used three different approaches for classification and generated results as mentioned in Figure 3. Approach proposed in this paper used machine learning for classification and three runs were submitted for the same. Runs submitted varied from each other in terms of classifiers used (Gaussian Naive Bayes, Logistic Regression and Random Forest Classifiers). Using the approach of Gaussian Naive Bayes classifier, an accuracy of 81.12 % was obtained, using Logistic Regression an accuracy of 80% was obtained and using Random Forest Classifiers an accuracy of 72.78% was obtained. The results in details, analysis and comparison for the same are discussed further.

6.1 Evaluation and Discussion

The MSIR, FIRE 2016, Subtask 1 organizer evaluated the results which gave a comparison of accuracy achieved by the 7 teams that participated as shown in Figure 3. The proposed approach (team BITS_PILANI) got ranked as 2nd with an accuracy of 81.12% for run1 while the highest accuracy achieved was 83.34% (by the team IINTU). Choice of Gaussian Naive Bayes classifier leads to the maximum accuracy attainment, as the proposed algorithm deals with the problem involving continuous attributes. Usage of Naive Bayes helps in building simplistic and highly scalable models which are fast and scale linearly with number of predictors and rows. Also the process of building a naive bayes model

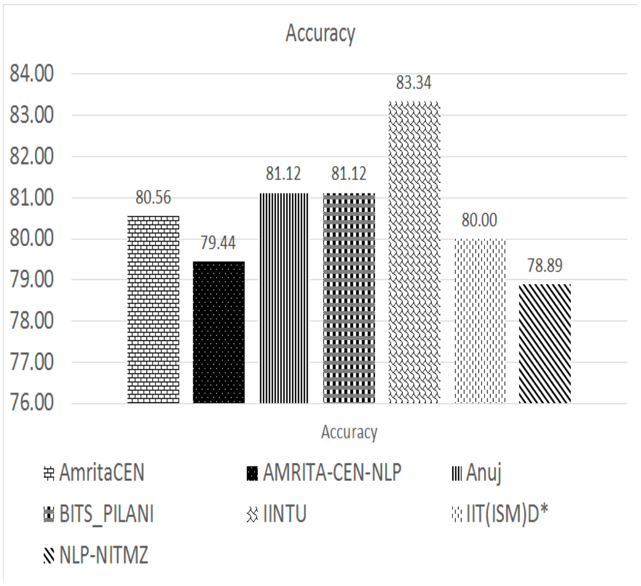


Figure 3: Highest Accuracy achieved by 7 teams that participated in MSIR,FIRE 2016

is highly parallelized even at the level of scoring. It was also observed from the results, that the proposed algorithm generated highest F-measure scores for the classes of Organization (ORG), Money (MNY) and Miscellaneous (MISC). Figure 4 shows the comparison of the different f-measure

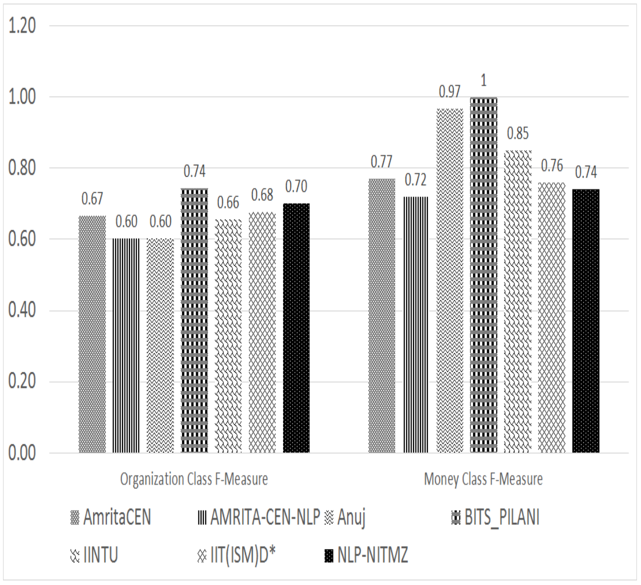


Figure 4: Comparison of F-Measure for Organization and Money class among different teams

scores of the teams obtained for the class Organization. The proposed algorithm (implemented by team BITS_PILANI) got the highest scores of 0.74418 using Gaussian Naive Bayes approach. This implies that the questions relating to a particular organization mainly being framed with words like ”which”, ”what” etc. could be efficiently classified by means of this approach. These scores can be attributed to the fact that the instances of the class ORG were maximum in the data set (67 out of 330 as discussed in Section 3). Also the proposed algorithm involves the formation of word level n-grams due to which words and phrases like ”which”, ”team”, ”series”, ”sponsor” etc. got associated, and thus might have contributed to an increase in the scores.

Figure 4 also shows the comparison of the different f-measure scores of the teams obtained for the class Money. Using the proposed algorithm (team BITS_PILANI) achieved the highest scores of 1 using Logistic Regression as a classifier (run 2). Hence all the questions relating to money being framed with words like ”how much”, ”price”, ”fare” etc. could be efficiently classified by means of the proposed approach. These high f-scores could be attributed to the efficient deployment of the word level n-gram techniques which in a way linked the words like ”fare”, ”how”, ”much”, ”price” etc. and thus might contributed to an increase in accuracy.

The evaluated results also showed that only two teams (team BITS_PILANI and team NLP-NITMZ) were able to identify instances belonging to Miscellaneous (MISC) class. This can be attributed to the fact that there were only 5 out of 330 instances of MISC class in the training data set. The proposed approach (team BITS_PILANI) got the highest scores of 0.2 using Gaussian Naive Bayes classifier, which again attributes for the simplistic approach of GaussianNB classifiers and the efficient deployment of the word level n-grams technique.

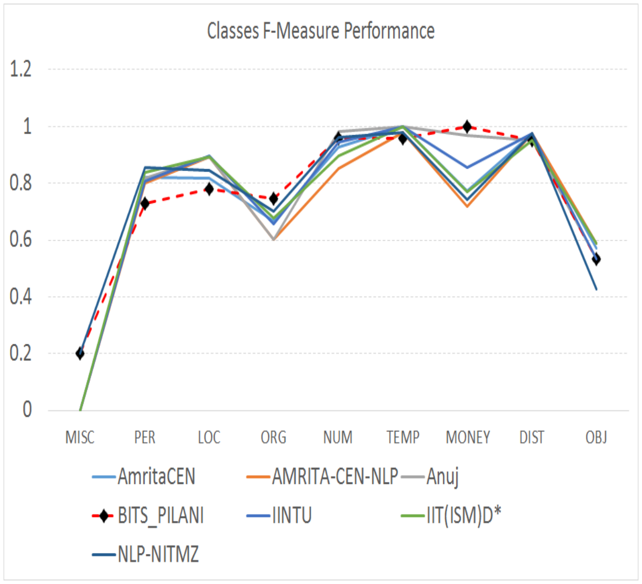
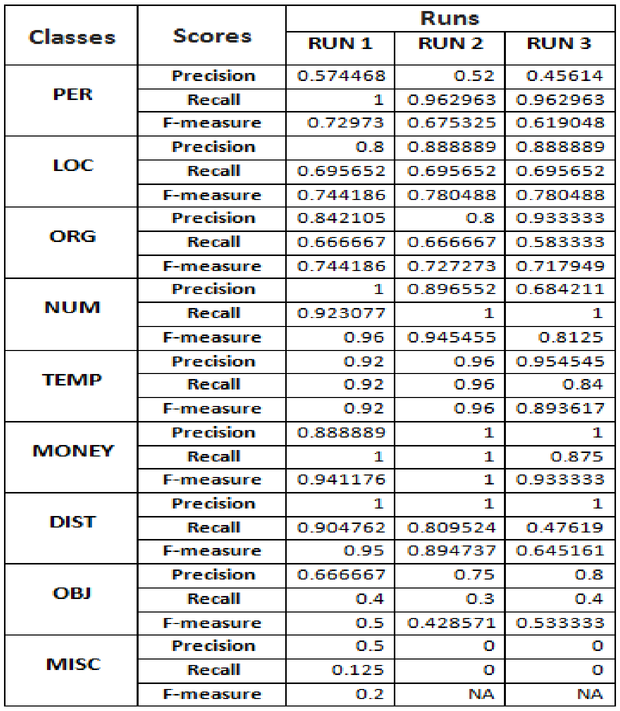


Figure 5: Comparison of F-Measure among different teams for various classes

Figure 5 shows a comparison of the accuracy obtained (taken the best accuracy obtained out of the three runs for each team) for classifying each of the nine classes. As evident from the figure, the proposed approach (implemented by team BITS_PILANI) was able to obtain satisfactory results in identifying the correct class labels particularly in the cases of MISC, ORG, MNY, NUM and OBJ classes with an f-measure score of 1 obtained for the class Money. Table 1 shows the scores of precision, recall and F-measure for each of the nine different classes, as evaluated by the FIRE 2016 task organizers [1], for the proposed algorithm (implemented by the team BITS_PILANI) for the three different runs submitted.

Table 1: Class wise score for all the runs submitted



Classes	Scores	Runs		
		RUN 1	RUN 2	RUN 3
PER	Precision	0.574468	0.52	0.45614
	Recall	1	0.962963	0.962963
	F-measure	0.72973	0.675325	0.619048
LOC	Precision	0.8	0.888889	0.888889
	Recall	0.695652	0.695652	0.695652
	F-measure	0.744186	0.780488	0.780488
ORG	Precision	0.842105	0.8	0.933333
	Recall	0.666667	0.666667	0.583333
	F-measure	0.744186	0.727273	0.717949
NUM	Precision	1	0.896552	0.684211
	Recall	0.923077	1	1
	F-measure	0.96	0.945455	0.8125
TEMP	Precision	0.92	0.96	0.954545
	Recall	0.92	0.96	0.84
	F-measure	0.92	0.96	0.893617
MONEY	Precision	0.888889	1	1
	Recall	1	1	0.875
	F-measure	0.941176	1	0.933333
DIST	Precision	1	1	1
	Recall	0.904762	0.809524	0.47619
	F-measure	0.95	0.894737	0.645161
OBJ	Precision	0.666667	0.75	0.8
	Recall	0.4	0.3	0.4
	F-measure	0.5	0.428571	0.533333
MISC	Precision	0.5	0	0
	Recall	0.125	0	0
	F-measure	0.2	NA	NA

6.2 Error Analysis

There are a few phases at which proposed approach could have attributed to the mis-classification of a few records. The proposed approach involves a dictionary based method for named entity recognition for which the corpus used had only limited entries due to which some of the entities might not have been recognized and removed. Also the data set had a large number of instances of named-entities which referred to the same name but had similar but different spellings. For instance, in the data set, words ”masjid” and ”mosjid” both referred to the same word implying ”mosque” but had different spelling. Since the proposed approach used a corpus for NER these entities couldn’t be removed unless all the spellings of these words were added to the corpus.

The proposed approach also involves the usage of a translation system (Google API[1]) for translating words of Bengali to English, but since the translation system did not consider the semantics of the sentence where the word was being used, it may have happened that the particular Bengali word would have been incorrectly translated. The given data set did not have a uniform distribution of class instances, as shown in Figure 1 the data set comprised only of 1.51% of MISC class instances while ORG class comprises 20% of the entries in the data set due to which the model trained could be biased. Also as mentioned before, not even a single instance of MISC class from the test data set could be identified by most of the teams, and even the proposed system was able to get an f-measure score of only 0.2 because of lesser number of instances of the class.

7. CONCLUSIONS AND FUTURE WORK

In this paper, a word-level n-gram based approach of classification of code mixed cross script question records into nine different coarse grained question type classes has been presented for Subtask 1 of MSIR, FIRE 2016. Presented approach uses a pipelined stages to classify questions using various machine learning algorithms(Gaussian Naive Bayes, Logistic Regression and Random Forest). Proposed approach obtained highest accuracy of 81.12% using Gaussian Naive Bayes approach among all the three runs submitted. Future work could be an improvisation of dictionaries for named-entity recognition for Bengali and English languages. Different Named Entity Recognizer and taggers along with trained models for Name Entity Recognition could be deployed. It would be interesting to find approaches by which implicit features about the code-mixed cross script data set could be efficiently trained using deep learning algorithms. Machine learning based models for language identification along with appropriate transliteration and translation tools (which take into consideration of the correct semantics) could be improved further.

References

[1] S. Banerjee, K. Chakma, S. K. Naskar, A. Das, P. Rosso, S. Bandyopadhyay, and M. Choudhury. Overview of the Mixed Script Information Retrieval (MSIR) at FIRE. In *Working notes of FIRE 2016 - Forum for Information Retrieval Evaluation, Kolkata, India, December 7-10, 2016*, CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[2] S. Banerjee, S. K. Naskar, P. Rosso, and S. Bandyopadhyay. The first cross-script code-mixed question answering corpus. In *First Workshop on Modeling, Learning and Mining for Cross/Multilinguality (MultiLingMine 2016) co-located with the 38th European Conference on Information Retrieval (ECIR 2016)*, volume 1589, pages 56–65, 2015.

[3] R. Bhargava, Y. Sharma, S. Sharma, and A. Baid. Query labelling for indic languages using a hybrid approach. In *Working notes of FIRE 2015 - Forum for Information Retrieval Evaluation, Gandhinagar, India, December, 2015*, volume 1587 of *CEUR Workshop Proceedings*, pages 40–42. CEUR-WS.org, 2015.

[4] S. N. Bhattu and V. Ravi. Language identification in mixed script social media text. In *Working notes of FIRE 2015 - Forum for Information Retrieval Evaluation, Gandhinagar,India, December, 2015*, volume 1587 of *CEUR Workshop Proceedings*, pages 37–39. CEUR-WS.org, 2015.

[5] M. Choudhury, P. Gupta, P. Rosso, S. Kumar, S. Banerjee, S. K. Naskar, S. Bandyopadhyay, G. Chittaranjan, A. Das, and K. Chakma. Overview of fire-2015 shared task on mixed script information retrieval. In *Working notes of FIRE 2015 - Forum for Information Retrieval Evaluation, Gandhinagar, India, December, 2015*, pages 19–25. CEUR-WS.org, 2015.

[6] X. Li and D. Roth. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

[8] D. Zhang and W. S. Lee. Question classification using support vector machines. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 26–32. ACM, 2003.