International Journal of Enhanced Research in Science, Technology & Engineering
ISSN: 2319-7463, Vol. 12 Issue 2, February-2023, Impact Factor: 7.957

Automated Semantic Role Labelling of Bengali-English Mixed Tweets

Sima Datta

Assistant Professor, Department of IT & CS, KNM Sonamura, Tripura India
Research Scholar, Department of CSE, NIT Agartala, Tripura, India

ABSTRACT

We demonstrate a technique for automatically labelling mixed Bengali-English tweets with Semantic Role Labeling. We investigate the problems caused by noisy, user-generated code-mixed social media data. We also compare the individual effects of different language characteristics in our system. Our suggested model is a two-step automated labelling approach that achieves an overall accuracy of 89% for Argument Classification, a 15% improvement over the existing rule-based baseline model. While there is relevant ongoing research on Semantic Role Labelling (SRL) and on building tools for code-mixed social media data, to the best of our knowledge, this is the first attempt at developing a statistical Semantic Role Labeler for Bengali-English code-mixed data.

Keywords: Semantic Role Label, Code Mix, Classification, Social Network, Natural Language Processing

INTRODUCTION

Semantic Role Labelling (SRL) is the process of recognizing and categorizing arguments of a particular predicate or verb in a phrase or utterance. These labels educate us about the role of the argument in relation to its predicate in the given phrase. With the increasing popularity of social media, there is an abundance of user-generated data available online on sites such as Facebook, Twitter, and Reddit, among many others. As a result, there is an increasing need to build tools to analyze this text in order to comprehend it. Bengali is an Indo-Aryan language spoken by 8.10% of India's total population, and it is also the official language of Bangladesh. The Eastern Nagari Script is the original script in which Bengali is written by natives. The intricacy required in blending numerous grammatical rules, and scripts, and the usage of transliteration in such code-mixed data is a significant difficulty for NLP applications. Thus, solving this problem becomes an increasingly significant task since a large portion of the material on social media exhibits this quality and will be of enormous benefit if mined[1].

Previous Work

The author [2] described several automated systems for corpus collection to collect code-mixed language data and phonetically translated. In this work, the authors have used a modified character n-gram with a weighted lexicon-based approach to obtain context information along with minimum weight. Code-mixing is a common occurrence in multilingual groups, both in everyday speech and on social media. The embedding of linguistic components such as phrases, words, and morphemes from one language into an utterance from another language is defined as code-mixing by [3]. Social media data, in particular Code-mixed text, does not precisely conform to the syntax, morphology, or structure of any of the participating languages, resulting in typical NLP tools failing to perform effectively with this data for a variety of tasks described their work in [4] [5].By conducting a comparative analysis of classifiers trained on several code-mixed characteristics [6]. Sophisticated approaches for learning sentiments in noisy code-mixed data utilizing sub-word LSTM have also been tested [7].For sentiment analysis on code-mixed data, [8] [9] tried binary polarity classification using several classes of supervised models [10].The authors [11] addressed automated semantic role-based labelling for Hindi-English mixed language on Twitter data. Semantic Role labelling for the Hindi language is described by [12].

For instance, the tweet is mixed Bengali and the English language is converted to the English language as shown below.

Tweet1: Thnks buds! ***Kabhi kabhi aajate acche*** photos

Translation: Thank you, buddy! Sometimes good photos are captured.

METHODOLOGY

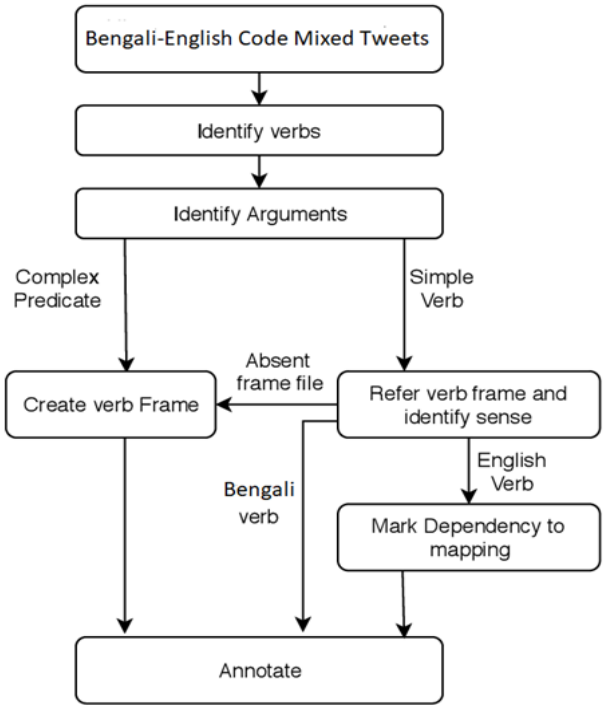

Figure1: Workflow of the Bengali-English Label Annotation

The labelling process described in figure1 is divided into two phases. The first stage is to identify an argument. Using the dependency tree structure, our model classifies all the tokens in the phrase as "Argument" or "Not an Argument." To do this, we label all direct dependents of the detected predicates as their Arguments, with the exception of tokens labelled as auxiliary verbs, post-positions, symbols (emojis in social media content), or those indicating coordination or subordination. The second phase is Argument Classification; in which we label the discovered arguments based on the previously indicated mappings. We add more rules to the modifier label mappings, and in the few circumstances when there is no such mapping, we train the model to label arguments with the most often occurring related label.

For binary classification, we employed Support Vector Models (SVM). This step's detected arguments are subsequently sorted into the various semantic roles listed in Table 1. For one-vs-rest multi-class classification, we employed the Linear SVC class of SVM [12]. The data was divided in an 80:20 split for training and testing. For training, all Linear SVC settings were set to default.

Features for code-mixed data

We wanted to examine what influence the specified language of a token would have because we are working with code-mixed text. As a result, we employed the following characteristics.

- Predicate + language: Predicate and its identified language.
- Headword + language: The chunk headword and its identified language.

RESULTS AND ANALYSIS

Individual aspects and their performance for the tasks of Argument Identification and Argument Classification are thoroughly examined. The accuracy, recall, and F1 scores of the characteristics for Argument Identification are shown in Table 1. The F1-score for Paninian Dependency labels is 85.

This research is carried out on Core i5 with 16GB Ram on a Colaboratory framework using Python programming. The data used for this experiment is BN-EN, taken from Twitter API. It consists of 1232 Bengali-English Code-mixed tweets comprising 12,345 tokens that are parsed and labelled with their semantic roles.

Argument Identification works well with Named Entities as well. This is due to the fact that Named Entities are often parameters to a predicate. They do not, however, provide much information on the role of the argument in the phrase. As a result, as seen in table 3, the score for Argument Classification is not particularly high.

Table 1: Individual feature performance for Argument Identification

Feature	Argument Identification		
	Precision	**Recall**	**f-score**
Predicate	38	57	44
Headword(HW)	55	47	53
Headword POS	38	50	46
Phrase type(PT)	46	34	42
Predicate-PT	49	65	53
Predicate-HW	59	49	53
Dependency	85	85	**85**
Named Entity	59	54	64
Headword POS-PT	46	40	39
Headword-PT	59	53	58
Headword POS(UD)	41	55	43
UDdependency	68	69	69
Predicate-language	47	66	55
Headword-language	59	52	57

We also see a significant increase in accuracy when we use the combinational feature of the predicate and its language, as compared to using only the predicate as a feature from above Table 1.Tweet2 and tweet3 are the examples from the corpus where the token “dekha”is the Bengali verb, ‘to see or watch’, the language of the predicate token can play an important role.

Tweet2: Irrfan Khan hollywood e abar dekha debe, trailer ta toh awesome ar acting o enjoyable.

Translation: Irrfan Khan will be seen in Hollywood again, trailer is awesome and acting is also enjoyable.

Tweet3: Ei movie take bar bar dekheo er matha mundu kichui bojha jaye na. Everything boddo confusing and amar mote not up to the mark.

Translation: After watching repeated times I can’t understand anything. Everything is so confusing and I think it’s not up to the mark.

Table2:Accuracy scores for Argument Identification.

Feature	Argument Identificatior		
	Precision	**Recall**	**f-score**
Baseline	65	63	59
with predicate-lang	59	62	58
+dependency	84	81	86

Table 2 displays the system's accuracy scores while employing baseline characteristics. When we utilize 'predicate language' as part of our baseline, the score does not change significantly. By adding a dependence label to our baseline features, we are able to get the maximum F1-score of 86 for this stage. The rule-based baseline model achieves 96.74% accuracy [14]. The basic model employs the sentence's dependency tree structure to identify the direct dependents of predicates as their arguments. Arguments do not include auxiliary verbs, postpositions, or symbols, among other things.

We used a hybrid method since the Classification phase is dependent on the identified reasons from the first stage. For Argument Identification, we employed a rule-based baseline system, and for Argument Classification, we used a statistical technique using SVM.

Table 3 shows the accuracy, recall, and F1 scores of the individual characteristics for Argument Classification. Paninian dependency labels again provide the best F1-score of 86. The UD dependency score is 80, which is somewhat lower. Tables 1 and 3 show that Paninian dependency labels performed better for both tasks. For both steps, there isn't much difference in performance between 'Headword POS' and 'Headword POS(UD)'.

Table 3: Individual feature performance for Argument Classification

Feature	ArgumentClassification		
	P	R	f-score
Predicate	06	09	06
Headword(HW)	18	10	13
Headword POS	05	07	06
Phrase type(PT)	08	10	08
Predicate-PT	05	08	06
Predicate-HW	05	06	06
Dependency	81	86	**86**
Named Entity	20	14	16
Headword POS-PT	07	09	08
Headword-PT	12	09	10
Headword POS(UD)	08	11	09
UDdependency	77	83	80
Predicate-language	06	10	07
Headword-language	18	11	14

The UD tagset is coarser in nature. The UD POS tagset comprises just 17 tags, compared to the 32 tags in the POS tagset produced for Indian languages [13]. Similarly, the Paninian dependency system contains 82 relations in total, whereas UD has just 40. We may conclude from the accuracy results that Paninian dependency labels capture more semantic information than UD dependency labels.

Table4: Accuracy scores for Argument Classification

Feature	Argument Classification		
	P	R	f-score
Baseline	27	15	19
+*dependency*	84	84	**89**

Table 4 shows the accuracy ratings for Argument Classification using baseline characteristics and with dependency labels. We got an F1 score of 89. This is a considerable increase above the baseline model's rule-based accuracy of 89% for Argument Classification [14].

CONCLUSION

In this paper, the authors have investigated the issues of code-mixed data on social media platforms. This research carried out the automatic labelling for Bengali-English code-mixed tweets by using Semantic Role Labelling with a hybrid approach named a rule-based and statistical method to identify and classify the arguments. The overall accuracy of this hybrid algorithm model is 89%.

REFERENCES

[1]. Soumil Mandal and Anil Kumar Singh. 2018. Language Identification in Code-Mixed Data using Multichannel Neural Networks and Context Capture. In Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text, pages 116–120, Brussels, Belgium. Association for Computational Linguistics.

[2]. Amitava Das and Björn Gambäck. 2014. Identifying Languages at the Word Level in Code-Mixed Indian Social Media Text. In Proceedings of the 11th International Conference on Natural Language Processing, pages 378–387, Goa, India. NLP Association of India.

[3]. Carol Myers-Scotton. 1997. Duelling languages: Grammatical structure in codeswitching. Oxford University Press.

[4]. Thamar Solorio and Yang Liu. 2008.Part-of-speechtagging for English-Spanish code-switched text.In*Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1051–1060.AssociationforComputationalLinguistics.

[5]. Ozlem Cetinoglu, Sarah Schulz, and Ngoc Thang Vu. 2016. Challenges of computational processing of code-switching. arXiv preprint arXiv:1610.02213.

[6]. Mandal, S. and Das, D. (2018). Analyzing roles of classifiers and code-mixed factors for sentiment

identification.arXiv preprint arXiv:1801.02581

[7]. Aditya Joshi, Ameya Prabhu, Manish Shrivastava and Vasudeva Varma. 2016. TowardsSub-Word Level Compositions for Sentiment Analysis of Hindi-English Code Mixed Text. Proceedingsof the 26th International Conference on Computational Linguistics (COLING 2016), pp. 2482–2491.

[8]. Ghosh, S., Ghosh, S., and Das, D. (2017a). Complexitymetric for code-mixed social media text. arXiv preprintarXiv:1707.01183.

[9]. Ghosh, S., Ghosh, S., and Das, D. (2017b). Sentimentidentification in code-mixed social media text. arXivpreprint arXiv:1707.01184.

[10]. Sharma, S., Srinivas, P., and Balabantaray, R. C. (2015) Text normalization of code mix and sentiment analysis. In Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on,pages 1468–1473. IEEE.

[11]. Maaz Anwar and Dipti Misra Sharma. 2016. Towards building semantic role labelers for Indian languages. In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016), pages 4588–4595.

[12]. FabianPedregosa,GaëlVaroquaux,AlexandreGram-fort,VincentMichel,BertrandThirion,OlivierGrisel, Mathieu Blondel, Peter Prettenhofer, RonWeiss, Vincent Dubourg, et al. 2011.Scikit-learn:Machine learning in python.*Journal of machinelearningresearch*,12(Oct):2825–2830.

[13]. Akshar Bharati, Rajeev Sangal, Dipti Misra Sharma,and Lakshmi Bai. 2006. Anncorra: Annotating corporaguidelinesforPOSandchunkannotationforIndianlanguages.*LTRC-TR31*,pages1–38.

[14]. Riya Pal and Dipti Misra Sharma. 2019. A dataset forsemantic role labelling of Hindi-English code-mixedtweets.In*Proceedingsofthe13thLinguisticAnnotationWorkshop*,pages178–188.