Efficient Multilingual Text Classification for Indian Languages

by

Salil Aggarwal, Sourav Kumar, Radhika Mamidi

Report No: IIIT/TR/2021/-1

Centre for Language Technologies Research Centre
International Institute of Information Technology
Hyderabad - 500 032, INDIA
September 2021

Efficient Multilingual Text Classification for Indian Languages

Salil Aggarwal
IIIT Hyderabad
salil.aggarwal@research.iiit.ac.in

Sourav Kumar
IIIT Hyderabad
sourav.kumar@research.iiit.ac.in

Radhika Mamidi
IIIT Hyderabad
radhika.mamidi@iiit.ac.in

Abstract

India is one of the richest language hubs on the earth and is very diverse and multilingual. But apart from a few Indian languages, most of them are still considered to be resource poor. Since most of the NLP techniques either require linguistic knowledge that can only be developed by experts and native speakers of that language or they require a lot of labelled data which is again expensive to generate, the task of text classification becomes challenging for most of the Indian languages. The main objective of this paper is to see how one can benefit from the lexical similarity found in Indian languages in a multilingual scenario. Can a classification model trained on one Indian language be reused for other Indian languages? So, we performed zero-shot text classification via exploiting lexical similarity and we observed that our model performs best in those cases where the vocabulary overlap between the language datasets is maximum. Our experiments also confirm that a single multilingual model trained via exploiting language relatedness outperforms the baselines by significant margins.

1 Introduction

Text classification is the task of assigning predefined categories to free-text documents with the use of natural language processing (NLP). Here, the classifier is fed a text and it returns a category based on the content. For the purpose of this paper, the task is to classify whether a piece of news article is regarding sports or not. This process of assigning tags or categories to text according to its content helps businesses automatically structure and analyze text quickly and cost-effectively to automate processes and enhance data-driven decisions. With the growth of the Internet around the world, users write comments in different languages. But the majority of current classification systems still address only a single language, mainly **English** (Dashtipour et al., 2016). This increases the risks of missing essential information in texts written in other languages. Also, training these systems require substantial amounts of annotated datasets, which is again an arduous task for many languages. Same is the case with Indian languages. Despite having a very large number of native speakers, most of the Indian languages are still considered to be resource poor. There are not enough datasets available in most of the domains. Therefore, the task of text categorization becomes challenging for Indian languages.

Therefore, in order to classify data in different languages, **multilingual text classification** techniques are the need of the hour. Training a multilingual model would refrain us from training a separate model for different languages and it also helps the system in better learning by means of parameter sharing. This approach mainly works by combining all the data in hand and studies in machine translation (MT) have shown that multilingual learning is not much efficient in case of unrelated languages (Kudugunta et al., 2019; Kunchukuttan and Bhattacharyya, 2020). But this is not the case with Indian languages. Underlying the vast diversity in Indian languages are many commonalities. Because of contact over thousands of years, most of the Indian languages have undergone convergence to a large extent (Sridhar, 1981). These languages share many common words which have the same **root word** and meaning. However, they use different scripts derived from the ancient **Brahmi script** (Kunchukuttan and Bhattacharyya, 2020), but correspondences can be established between equivalent characters across different scripts. So, the main question arises whether we can benefit

from the relatedness found in between Indian languages? By relatedness, we refer to languages that exhibit lexical and structural similarities on account of sharing a common ancestry.

Thus, in this work, we put our efforts in exploring the zero-shot as well as multilingual text classification via exploiting lexical similarity of Indian languages. For zero-shot classification, we are proposing the efficient way of reusing a classification model trained on one language on some other Indian language. In addition to this, we also tackled the problem of deciding which language model to use for zero-shot classification for a particular test language. Our results confirm that maximum accuracy is achieved in those cases where the vocabulary overlap between the two language datasets is maximum. For efficient multilingual text classification, we are exploiting the lexical similarity via two techniques namely **unified transliteration** and **subword segmentation**. Our experiments also confirm that in case of low resource related languages, multilingual models achieve better accuracy than the baseline models.

This paper is further divided into 5 sections. Section 2 discusses related work in this area. Section 3 elaborated the methodology behind the different techniques and experiments. Section 4 elaborates the experimental details including the dataset preparation, dataset pre-processing and the experimental setup for training our models. All the results and analysis have been discussed in Section 5. Section 6 talks about conclusion and possible future work.

2 Related Work

One of the main problems in multilingual classification is the significant lack of resources (Balahur and Turchi, 2012). Thus, analysis in multiple languages is often addressed by transferring knowledge from resource-rich to resource-poor languages (Denecke, 2008), because there are no resources available in other languages. Much of the work in subjectivity analysis has been applied to English data, though work on other languages is growing: e.g., Japanese data are used in (Kobayashi et al., 2004; Suzuki et al., 2006; Kanayama and Nasukawa, 2006), German data are used by Kim and Hovy (2006b). Lexical approaches for sentiment analysis necessitate language specific lexical and linguistic resources. Generating these resources is very time consuming and often requires a lot of manual work. Methods have been developed for the mapping of subjectivity lexicons to other languages. To this aim, Kim and Hovy (2006a) use a machine translation system and subsequently use a subjectivity analysis system that was developed for English to create subjectivity analysis resources in other languages.

Another approach in obtaining subjectivity lexicons for other languages than English was explored by Banea et al. (2008b). In this work, authors attempt to leverage on the resources available for English and, by employing machine translation, generate resources for subjectivity analysis in other languages. This paper introduces a method for creating a subjectivity lexicon for languages with scarce resources. Further on, another approach to building lexicons for languages with scarce resources is presented by Banea et al. (2008a). This method is able to build a subjectivity lexicon by using a small seed set of subjective words, an online dictionary, and a small raw corpus, coupled with a bootstrapping process that ranks new candidate words based on a similarity measure. Machine translation for multilingual text classification has also seen attention from researchers. The approach is to use a machine translation system to translate texts in other languages into English : the text is translated from the original language into English, and then English-language resources such as SentiWordNet are employed (Denecke, 2008). Kanayama et al. (2004) translated only sentiment units with a pattern based approach.

Balahur and Turchi (2014) used uni-grams, bi-grams and tf-idf features for building support vector machines on translated text. Boyd-Graber and Resnik (2010) built Latent Dirichlet allocation models to investigate how multilingual concepts are clustered into topics. Translation systems, however, have various problems, such as sparseness and noise in the data (Balahur and Turchi, 2012). Sometimes, the translation system does not translate essential parts of a text, which can cause serious problems, possibly reducing well-formed sentences to fragments. Therefore, we put our efforts for training a multilingual classifier without

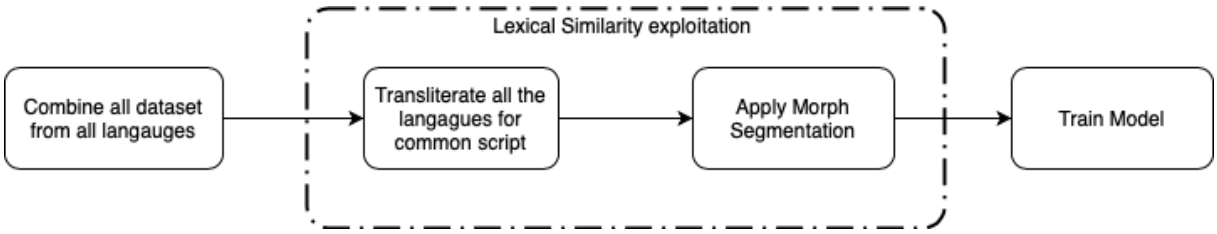

Figure 1: Multilingual Text Classification Pipeline

the use of existing machine translation systems.

3 Methodology

India is known as the land of many tongues (Kunchukuttan and Bhattacharyya, 2020). There is no single language called "Indian". India speaks hundreds of languages and dialects (Sengupta and Saha, 2015). Some are extinct, while some are still in use with considerable speakers. Despite having a lot of different scripts, most of the Indian languages still share a lot of lexical features and common words which have the same root and meaning which can be utilized to help improve the quality of zero-shot as well as text classification systems trained on them. So, in this paper we propose our technique of performing zero-shot classification via exploiting the language relatedness. Also, we investigate how multilingual classification models perform in case of Indian languages. To do this efficiently, we exploited the lexical similarity via two techniques namely **unified transliteration and subword segmentation**.

3.1 Exploiting Lexical similarity

Unlike the original multilingual text classification techniques which mostly aim at transfer learning via parameter initialisation i.e. learning from one high resource language and then transferring knowledge to some low resource language, we are exploiting lexical similarity between related Indian languages via parameter sharing. For this, we combined the two different approaches namely unified transliteration and subword segmentation to ensure that there is sufficient overlap between the vocabularies of the related Indian languages datasets.

3.1.1 Unified Transliteration

Since the languages involved in the models have different scripts, the data processing should help to map them into a common single script. So here, we transliterate all the Indian languages into a common **Devanagari script** (which in our case is script for Hindi) to share the same **surface form** (Kunchukuttan and Bhattacharyya, 2020). This unified transliteration is a string homomorphism, replacing characters in all the languages mentioned above with Hindi characters (script conversion to Devanagari) or consonant clusters independent of context.

3.1.2 Subword Segmentation

Despite sharing a lot of cognates, Indian languages do not share many words at their non-root level. Therefore, the more efficient approach is to exploit Indian languages at their sub-word level which will ensure more vocabulary overlap. Therefore, we are converting every word to sub-word level using the very well known technique Byte Pair Encoding (BPE) (Sennrich et al., 2015). This technique is applied after the unified transliteration in order to ensure that languages share same surface form (script). BPE units are variable length units which provide appropriate context for translation systems involving related languages. Since their vocabularies are much smaller than the morpheme and word level models, data sparsity is also not a problem. In a multilingual scenario, learning BPE merge rules will not only find the common sub-words between multiple languages but it also ensures consistency of segmentation among each considered language pair.

3.2 Zero-Shot Text Classification

There are many languages in India and one can not expect annotated data available in all of the the domains for all of the languages. So in zero-shot text classification, the model can classify any text between given labels without any prior training data. For performing it efficiently for Indian languages, we are using the vocabulary overlap technique as discussed in **Section 3.2.1**. From our experiments, we noticed that the zero-shot classification performs best in those cases where the vocabulary overlap is **maximum** between the different language datasets. That model will perform best on that language, which is most similar to the

Language	pa	gu	mr	or	bn	ta	te	ml	kn
pa	-	67.87	**71.34**	58.35	55.77	38.55	61.27	54.16	61.87
gu	41.54	-	**88.85**	70.84	61.21	46.45	75.02	65.60	75.90
mr	18.32	37.28	-	35.08	28.36	55.34	70.35	59.27	**76.14**
or	32.27	64.02	**75.53**	-	60.30	50.21	67.46	67.35	69.57
bn	42.77	76.70	**84.69**	83.61	-	46.89	70.98	66.43	75.45
ta	11.75	23.13	65.68	27.67	18.63	-	75.23	**79.43**	77.24
te	18.46	36.93	82.52	36.74	27.88	74.35	-	82.00	**92.77**
ml	14.11	27.92	60.12	31.73	22.57	67.89	70.91	-	**77.81**
kn	11.80	23.65	56.54	23.99	18.76	48.33	**58.73**	56.97	-

Table 1: Vocabulary Overlap Matrix

training language dataset using vocabulary overlap technique. The method of calculating vocabulary overlap matrix is explained in the further subsection.

3.2.1 Vocabulary Overlap

We calculated similarity scores for each language based on vocabulary overlap by considering other language training data as monolingual data. Vocabulary Overlap provides a crude measure of surface form similarity between two languages. It is efficient to calculate, and is often quite effective, especially for low-resource languages. Here, we use the number of tokens that two languages share to measure the similarity between them. First, we transliterated each language to a common script, in our case it is Devanagari script and then all the tokens were converted into their respective subwords. After that score was calculated using the percentage of the common tokens in the two languages as:

$$vocab_{(l1-l2)} = \frac{|Token_{l1} \cap Token_{l2}|}{|Token_{l1}|} * 100$$

The details of the vocabulary overlap matrix is shown in **Table 1**.

3.3 Multilingual Text Classification

We trained a Multilingual model, which is a single model that can handle multiple languages simultaneously. This would circumvent having to train a monolingual model for every single Indian language. One example in our case would be to classify whether a piece of news article is regarding sports or not. Using a regular Machine learning or Deep learning model, we would be able to classify only Punjabi language sports articles but not articles written in other Indian languages say Marathi. But if we use a multilingual model, we would be able to classify news articles in Punjabi, Marathi and multiple other Indian languages. Our results also prove that multilingual models achieve better performance than monolingual models, especially for low-resource languages. This could be the ability to learn not just from the training data of the language in question, but also from other language datasets.

But we noticed that this learning is not much efficient in case of the languages that don't show any kind of relatedness. But on the other hand, Indian languages exhibit a lot of lexical and structural similarities on account of sharing a common ancestry. It is therefore important to utilize the lexical similarity of these languages to build systems by combining all the related languages. For the scope of this paper, we have trained 2 kinds of multilingual models. One with the training data of all the languages combined in their respective scripts and the other in which all the languages are first transliterated into one common script, in our case in Devanagari and then they are combined. To the best of our knowledge, this is the first time when someone has exploited the lexical similarity found in between Indian languages by using the techniques of unified transliteration and subword segmentation for the task of text classification. The pipeline is shown in **Figure 1**.

4 Experiments

4.1 Dataset

We have used **IndicNLP News Article Classification Dataset** (Kunchukuttan, 2020) for performing our experiments. This classification dataset comprises news articles and their categories for 9 different Indian languages. The dataset is balanced

Language	pa	gu	mr	or	bn	ta	te	ml	kn
pa	-	64.88	**72.97**	56.69	54.96	55.28	56.75	55.13	57.79
gu	64.78	-	**74.10**	61.44	61.51	54.31	66.56	60.19	68.00
mr	58.60	60.66	-	59.20	59.46	55.60	64.88	53.78	**69.29**
or	58.53	69.43	**74.43**	-	65.06	46.34	52.57	55.28	70.53
bn	49.40	56.47	**63.79**	56.22	-	47.13	54.91	51.68	53.24
ta	46.03	54.44	59.28	52.72	55.78	-	57.82	**64.96**	55.49
te	58.31	58.53	62.47	57.57	60.21	61.03	-	59.04	**74.65**
ml	52.60	56.03	58.50	55.10	62.44	58.51	63.21	-	**75.78**
kn	56.13	69.49	69.49	63.19	65.62	60.06	**72.63**	58.52	-

Table 2: Results of Zero-Shot Text Classification

Model	pa	gu	mr	or	bn	ta	te	ml	kn
Baseline	93.69	95.01	93.97	95.45	94.26	95.82	96.05	92.48	94.15
Multilingual without Transliteration	95.82	96.66	95.84	96.77	95.23	97.17	96.94	94.36	94.72
Multilingual with Transliteration	**97.66**	**98.77**	**97.49**	**97.10**	**96.46**	**97.77**	**98.33**	**95.83**	**96.99**

Table 3: Experimental results of all compared models.

across different classes. There were a lot of classes that were not common to all the languages. Since the task in hand was to classify whether a piece of news article is regarding sports or not, we discarded all the other classes other than sports. For creating this dataset, negative examples were uniformly taken from the rest of the classes. The new dataset contains 1360 training and 160 test examples for each language. The languages used in our dataset are Punjabi (pa), Gujarati (gu), Marathi (mr), Odia (Or), Bangla (bn), Tamil (ta), Telugu (te), Malayalam (ml) and Kannada (kn).

4.2 Dataset Preprocessing

We noticed that the dataset contains a lot of punctuation, so we used manually created regex for cleaning the entire corpora. For all of our our experiments, we have taken an equal number of training sentences from all the languages in order to maintain uniformity while training multilingual models. We have used the Indic NLP library(Kunchukuttan, 2020) for tokenization and normalization as our pre-processing steps. Also, to make sure that all the languages share the same surface form and have sufficient vocab overlap, we again used Indic NLP library (Kunchukuttan, 2020) for unified transliteration and subword segmentation.

4.3 Training Details

The dataset was split into a ratio of 4:1 for the purpose of training and testing. All of our experiments were performed using **5-fold cross-validation**. We used an embedding layer followed by a Bidirectional LSTM layer followed by a dense layer. The embedding size was set to 128 and 128 hidden units were used in the LSTM layer. All the models were trained for 20 epochs with a batch size of 64. All of the experiments were performed using Keras library (Chollet et al., 2015) with Tensorfow (Abadi et al., 2015) as its backend.

5 Results and Analysis

Table 2 shows the results of zero-shot text classification on various languages. We can see that the maximum accuracy is achieved in those cases where the vocabulary overlap between the training and the monolingual dataset is maximum. For eg, Marathi model will perform best on the Punjabi test set due to maximum overlap. This can be confirmed from our vocabulary overlap matrix shown in **Table 1**. The reason is that most of the Indian languages share a lot of common sentiment bearing words and we exploit this using converting everything into same script and every word to its component subwords.

Table 3 shows the results of baselines as well as our multilingual models. We can see from the table that both the multilingual models outperform the baseline models. The reason is the ability to learn not only from its own dataset, but also from the other related languages present in the dataset. The best accuracy is achieved in the case where we apply the technique of unified transliteration and subword segmentation. The reason as explained above is the increase in vocabulary overlap which further increases the accuracy as compared to the first case. So, we can observe that in both the cases, we have benefitted from the lexical similarity found in Indian languages.

6 Conclusions and Future Work

In this work, we explored effective methods to exploit lexical similarity between related Indian languages in order to improve the quality of classification systems on low resource Indian languages. Our results confirm that a model trained on one Indian language can be reused for another Indian language and will provide good results if there is a significant vocabulary overlap between the two datasets. We also proved that for low resource Indian languages, multilingual models outperform the baseline models. The reason is the ability to learn not just from the training data of the language in question, but also from the other language datasets. Unified Transliteration and Subword Segmentation further increase the accuracy by increasing the vocab overlap among the datasets. Also, to get more increase in accuracy, one can try with large size datasets. For future work, we will try to apply this technique to other NLP related tasks for Indian languages.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Alexandra Balahur and Marco Turchi. 2012. Multilingual sentiment analysis using machine translation? In *Proceedings of the 3rd workshop in computational approaches to subjectivity and sentiment analysis*, pages 52–60.

Alexandra Balahur and Marco Turchi. 2014. Comparative experiments using supervised learning and machine translation for multilingual sentiment analysis. *Computer Speech & Language*, 28(1):56–75.

Carmen Banea, Rada Mihalcea, and Janyce Wiebe. 2008a. A bootstrapping method for building subjectivity lexicons for languages with scarce resources. In *LREC*, volume 8, pages 2–764. Citeseer.

Carmen Banea, Rada Mihalcea, Janyce Wiebe, and Samer Hassan. 2008b. Multilingual subjectivity analysis using machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 127–135.

Jordan Boyd-Graber and Philip Resnik. 2010. Holistic sentiment analysis across languages: Multilingual supervised latent dirichlet allocation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 45–55.

Francois Chollet et al. 2015. Keras.

Kia Dashtipour, Soujanya Poria, Amir Hussain, Erik Cambria, Ahmad YA Hawalah, Alexander Gelbukh, and Qiang Zhou. 2016. Multilingual sentiment analysis: state of the art and independent comparison of techniques. *Cognitive computation*, 8(4):757–771.

Kerstin Denecke. 2008. Using sentiwordnet for multilingual sentiment analysis. In *2008 IEEE 24th international conference on data engineering workshop*, pages 507–512. IEEE.

Hiroshi Kanayama and Tetsuya Nasukawa. 2006. Fully automatic lexicon expansion for domain-oriented sentiment analysis. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 355–363.

Hiroshi Kanayama, Tetsuya Nasukawa, and Hideo Watanabe. 2004. Deeper sentiment analysis using machine translation technology. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 494–500.

Soo-Min Kim and Eduard Hovy. 2006a. Automatic identification of pro and con reasons in online reviews. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 483–490.

Soo-Min Kim and Eduard Hovy. 2006b. Identifying and analyzing judgment opinions. In *Proceedings of the human language technology conference of the NAACL, main conference*, pages 200–207.

Nozomi Kobayashi, Kentaro Inui, Yuji Matsumoto, Kenji Tateishi, and Toshikazu Fukushima. 2004. Collecting evaluative expressions for opinion extraction. In *International Conference on Natural Language Processing*, pages 596–605. Springer.

Sneha Reddy Kudugunta, Ankur Bapna, Isaac Caswell, Naveen Arivazhagan, and Orhan Firat. 2019. Investigating multilingual nmt representations at scale. *arXiv preprint arXiv:1909.02197*.

Anoop Kunchukuttan. 2020. The IndicNLP Library. https://github.com/anoopkunchukuttan/indic_nlp_library/blob/master/docs/indicnlp.pdf.

Anoop Kunchukuttan and Pushpak Bhattacharyya. 2020. Utilizing language relatedness to improve machine translation: A case study on languages of the indian subcontinent. *arXiv preprint arXiv:2003.08925*.

Debapriya Sengupta and Goutam Saha. 2015. Study on similarity among indian languages using language verification framework. *Advances in Artificial Intelligence*, 2015.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*.

SN Sridhar. 1981. Linguistic convergence: Indo-aryanization of dravidian languages. *Lingua*, 53(2-3):199–220.

Yasuhiro Suzuki, Hiroya Takamura, and Manabu Okumura. 2006. Application of semi-supervised learning to evaluative expression classification. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 502–513. Springer.