Introduction

Plagiarism has become a serious concern in science, academia and journalism. The number of scientific publication retractions have risen sharply in the last few years. With ever increasing number of articles being published, it has become a challenge to detect plagiarism. Automated detection of monolingual plagiarism has been explored in-depth in recent years [1][2][3][4][5]. However, plagiarism detection across multiple languages has been a largely underexplored area.

Cross-Lingual Plagiarism Detection (CLPD) is defined as detecting parts of a given document that “match”, exactly or semantically, parts of a given set of documents in a different language. Within CLPD, we tackle the subproblem of "**cross lingual paraphrase detection**" - detecting whether a sentence in English is a paraphrase in Hindi. The problem is also called translation plagiarism in literature [15] and it is typically solved by creating a translated representation of the input sentence in English and comparing the representation with the sentence in Hindi. We discuss various ways to do the translation and provide a new approach using sentence embeddings. The models are evaluated on the Bible and Tides multilingual corpus. The proposed model performs close to the existing state-of-art approaches. We hope this work will spur on research on sentence vector representation for cross lingual plagiarism detection.

Method

Materials

We use two different datasets throughout the project. The first one has 30,000 verses from the Bible in both English and Hindi. This data is extracted from [6], a massively parallel corpus which includes verses from the Bible in 100 different languages. The second one is the Tides corpus [7] which includes 50,000 English-Hindi sentence pairs.

For the baseline model where we need word translations from English to Hindi, we use an offline dictionary [8] that is extracted from Wikipedia's translations section. For models that are based on IBM-M1, we use NLTK's POS tagger [9] for English sentences and RDR-POSTagger for Hindi sentences [10][11].

Standard implementation of the state of the art method, CLPD - Alignment Similarity Analysis [16] is used. It proposes a variant of standard statistical translation technology based on the IBM-M1 alignment model to calculate the similarity between two sentences across different languages.

$$\varphi\left(d_f, d_e\right) = \varrho(d_e)\, \omega\left(d_f \middle| d_e\right)$$

Where,
ω is translation model trained using IBM M1.
φ is length model that penalizes sentences that are of different lengths.

We also use standard implementation of word2vec [17][18][19] for getting vector representations of words in the training data.

Procedure

The main objective is to retrieve candidate sentences in Hindi which are likely to be paraphrases of the input sentence in English. To achieve this we create a "Hindi representation" of the sentence in English by considering it as a bag of words.

VectorSim

We propose a semantic embedding based approach by learning a projection or transfer function between vector spaces that represent each language. This approach is inspired from the work by Mikolov [13] for machine translation. The methodology consists of two main steps. First, a model for the vector representation of each sentence in each language is developed. This part is repeated separately for each language and the outputs from this part are two distinct models providing the vector representation of a sentence in each language. Second, a transformation between sentence embeddings in English and Hindi is found using a linear regression model. The methodology proposed in this work is illustrated in Figure 1.

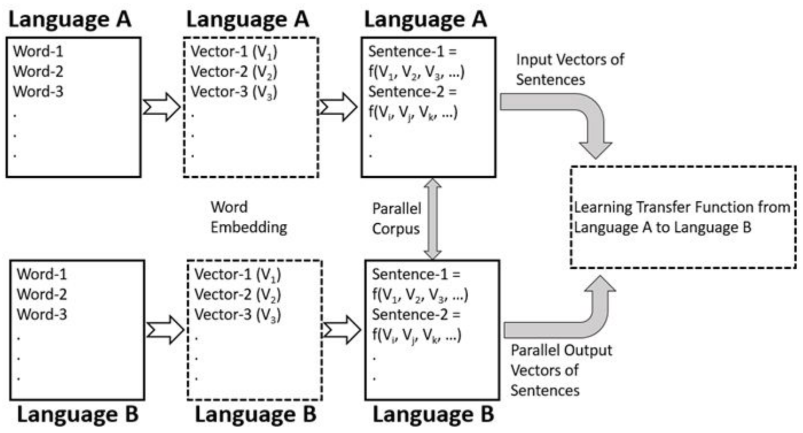
Language A
Word-1
Word-2
Word-3
Language A
Vector-1 (V1)
Vector-2 (V2)
Vector-3 (V3)
Language A
Sentence-1 = f(V1, V2, V3, ...)
Sentence-2 = f(Vi, Vj, Vk, ...)
Input Vectors of Sentences
Word Embedding
Parallel Corpus
Learning Transfer Function from Language A to Language B
Word-1
Word-2
Word-3
Vector-1 (V1)
Vector-2 (V2)
Vector-3 (V3)
Sentence-1 = f(V1, V2, V3, ...)
Sentence-2 = f(Vi, Vj, Vk, ...)
Parallel Output Vectors of Sentences
Language B
Language B
Language B

Figure 1: Methodology flowchart showing different steps required in the proposed method. Dashed boxes show parts for machine learning models.

Transfer model allows to compare two sentences in different languages by transferring a vector from vector space A to vector space B. The developed model is used to transfer the sentence vector in English to the space in other language. Mean squared error is computed between the transferred vector and all candidate sentences in Hindi. The sentences can be then ranked based on their squared error to the transferred vector where the minimum distance is the closest sentence to the sentence in English.

Representation of Sentences

There are various methods proposed in literature for predicting the vector representation of a sentence or a phrase in one language. Word embedding as distributed representations of words [17][18][19] is used to create word level representation and a sentence vector is created by taking sum of all word vectors for a whole sentence or phrase.

Vector Space Mapping between two languages

Once the vector representation of each sentence in a language is calculated, a regression model is built on a parallel corpus with vectors of each sentence in English as input and vectors of sentences in Hindi as output.

$$\min \left\| A.\overrightarrow{d_e} - \overrightarrow{d_f} \right\|$$

A linear regression model is used to provide vector transformation between the two vector spaces. Once the vector for a sentence in English is transferred using the regression model, the distance or similarity score between this vector and other vectors representing sentences in Hindi can be measured. The R^2 for the regression was 0.60 on the development set and 0.57 on the test set for both datasets.

Evaluation

Baseline

We propose two baselines that create a representation of the English sentence by translating words from a given dictionary. In the first baseline, we convert the English sentences by replacing their words with their first translations in an offline dictionary.

English Sentence: **Sun rises in the east**
Word by word translation: **[सूरज, उगता, में,है, पूर्व]**
Word by word transliteration (for understanding): **[Sooraj, ugta, me, hai, purva]**
Hindi Sentence: **सूर्य पूर्व में उगता है**

To improve on the existing baseline, we trained IBM-M1 alignment model for words across sentences in English and Hindi. Few instances are shown in Table 1. We replaced the standard dictionary in the first baseline with the dictionary found by the alignment model. For both these approaches, we calculated the Jaccard similarity between the word by word translated sentence and actual sentence in Hindi.

English word	**Aligned Word from corpus**	**Transliteration (for understanding)**
temple	मंदिर	Mandir
thee	तुझे	Thuje
gold	सोना	Sona
thereof	इनकी	Inki

Table 1: Aligned words learned using IBM-M1 Alignment model from the training data.

We compare the different translation based approaches by defining the problem of paraphrase detection as a retrieval problem across English and Hindi. Formally, given a sentence S in English we retrieve top-k sentences in Hindi, in the test set, that is the most similar to the sentence S. We split the dataset with 99% for training, 0.5% for tuning and 0.5% for testing. We are calculating the retrieval score at top-1,3,5,10 sentences and reporting them in Table 2. The accuracy is calculated using the following ratio of number of hits by total number of sentences. That is, if the actual paraphrased sentence is in the top-k result reported by the algorithm.

$$accuracy = \frac{\#hits}{\#total}$$

We have tuned the number of dimensions for the word vectors using a development set. The result of the tuning is given in Figure 3.

Results

The results of the baseline approaches, CLPD-ASA and the proposed VectorSim approach has been given in the Table 2.

Dataset	**Algorithm**	**Top-1**	**Top-3**	**Top-5**	**Top-10**

Bible	Baseline - Jaccard-Standard	0.154	0.246	0.303	0.366
	Baseline - Jaccard-Alignment	0.206	0.343	0.360	0.383
	CLPD-ASA (State of the Art)	**0.491**	**0.714**	**0.794**	**0.840**
	VectorSim (Our Approach)	0.458	0.646	0.720	0.791
Tides	Baseline - Jaccard-Standard	0.268	0.348	0.408	0.488
	Baseline - Jaccard-Alignment	0.332	0.484	0.512	0.520
	CLPD-ASA (State of the Art)	**0.604**	**0.752**	**0.800**	**0.844**
	VectorSim (Our Approach)	0.342	0.534	0.584	0.688

Table 2: Accuracy of our approach compared with baselines and state of the art.

Dataset	**μ**	***σ***
Bible	1.025	0.22
Tides	1.108	0.26

Table 3: Mean and standard deviation of ratio of sentence lengths in training data.

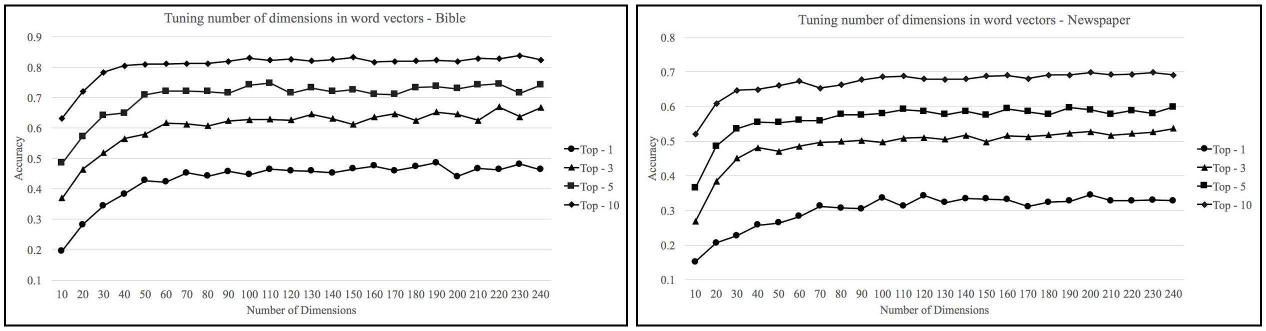

Figure 3: Tuning the parameters for number of dimensions across two datasets (a) Bible (b) Tides. Both achieve saturation after 150 dimensions. We have chosen 200 dimensions as it give majority maximum across all top-k accuracy measures

Discussion

The approaches discussed in this paper are: a) Jaccard-Standard, b) Jaccard-Alignment, c) CLPD-ASA and d) VectorSim. Based on the results shown in Table 2, it is observed that CLPD-ASA method is performing better than all other approaches. This means that finding the word alignment in the other language is not limited only to the dictionary first retrieved or most probable word used in Jaccard-Alignment and Jaccard-Standard respectively.

The disadvantage of Jaccard-Standard is the fact that it only considers the first word retrieved from the dictionary used as database. Some words in Hindi sentence that carry information and do not have a corresponding word in the English sentence are not used in the similarity analysis.

In Jaccard-Alignment, we select the highest probable translated word for every word in English. However, since there are several candidate words in the target language for a word from English, there is a low chance that for even the highest probable word to occur in the corresponding sentence in Hindi. This decreases the overall jaccard similarity score. The two approaches are very simple to implement and act as a good baseline with around 38% accuracy for Bible and 50% accuracy for Tides dataset.

VectorSim outperforms the defined baseline algorithms. There are two main outcomes of this approach:
(i) Vectors of words in one language is a linear transformation into another language. We hope transformation learned using the English-Hindi parallel corpus will pave way for further research into study of transfer learning in word, sentence and document embeddings. We have achieved comparable performance with the state of the art.
(ii) Embeddings for a sentence can be generated by sum of all the word vectors in the sentence. We have also tried other approaches like average of the vectors and weighted sum using TF-IDF, but a simpler approach was better than the rest.
Considering more than 50 dimension for the vectors representing each word and then sentence is bringing more aspects and information embedded inside the sentences. Moreover, the word embedding vectors for similar and close words have lower distance compared to the vectors of other irrelevant words. This means that the model is not limited by a single word for alignment which is the drawback of both Jaccard-Alignment and Jaccard-Standard baselines. This issue is avoided in VectorSim model since there is no limitation for word embedding vectors and the ones with similar meaning in each language are highly similar.

The VectorSim model outperforms the baseline approaches, but does not beat the performance of state of the art. The main difference is the part related to word embedding vectors in VectorSim versus word-to-word alignment consideration in CLPD-ASA. Training of the alignment model considers alignment between the two sentences and learns it together. The word vectors of the proposed VectorSim approach are learned separately for each language and the linear transformation learns the proper mapping function between the two languages vector spaces. One of the challenges of the proposed method is finding an accurate transformation between the two spaces. This becomes even more challenging as the dimensionality of vector representations increases. This is another reason why our model falls short compared to CLPD-ASA.

Although the sentence lengths are considered in several algorithms explored in this study, none of them consider the sequence of the words in a sentence. Using RNN models to find the vector representation at sentence level will be an improvement. The same word embedding models can be used for retrieving vector representation of the words. However, in the next step, instead of using sum to find the vector representation of a sentence, an RNN model will help to find a more representative vector considering sequence of words in the sentence as well.

References

1. Parth, Gupta, Rao Sameeer, and Prasenjit Majumdar. "External plagiarism detection: N-gram approach using named entity recognizer." *CLEF (Notebook Papers/LABs/Workshops)*. 2010.
2. He, Hua, Kevin Gimpel, and Jimmy J. Lin. "Multi-Perspective Sentence Similarity Modeling with Convolutional Neural Networks." *EMNLP*. 2015.
3. Alzahrani, Salha M., Naomie Salim, and Ajith Abraham. "Understanding plagiarism linguistic patterns, textual features, and detection methods." *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42.2 (2012): 133-149.
4. Zu Eissen, Sven Meyer, Benno Stein, and Marion Kulig. "Plagiarism detection without reference collections." *Advances in data analysis*. Springer Berlin Heidelberg, 2007. 359-366.
5. Potthast, Martin, et al. "An evaluation framework for plagiarism detection." *Proceedings of the 23rd international conference on computational linguistics: Posters*. Association for Computational Linguistics, 2010.

6. Christodouloupoulos, C., & Steedman, M. (2015). A massively parallel corpus: the bible in 100 languages. *Language resources and evaluation*, *49*(2), 375-395.
7. Bojar, O., Diatka, V., Rychlý, P., Straňák, P., Suchomel, V., Tamchyna, A., & Zeman, D. (2014, May). HindEnCorp-Hindi-English and Hindi-only Corpus for Machine Translation. In *LREC* (pp. 3550-3555).
8. https://en.wiktionary.org/w/index.php?title=User:Matthias_Buchmeier
9. Bird, S. (2006, July). NLTK: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions* (pp. 69-72). Association for Computational Linguistics.
10. Dat Quoc Nguyen, Dai Quoc Nguyen, Dang Duc Pham and Son Bao Pham. RDRPOSTagger: A Ripple Down Rules-based Part-Of-Speech Tagger. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, EACL 2014, pp. 17-20, 2014.
11. Dat Quoc Nguyen, Dai Quoc Nguyen, Dang Duc Pham and Son Bao Pham. A Robust Transformation-Based Learning Approach Using Ripple Down Rules for Part-Of-Speech Tagging. *AI Communications* (AICom), vol. 29, no. 3, pp. 409-422, 2016.
12. Zu Eissen, Sven Meyer, Benno Stein, and Marion Kulig. "Plagiarism detection without reference collections." *Advances in data analysis*. Springer Berlin Heidelberg, 2007. 359-366.
13. Mikolov, Tomas, Quoc V. Le, and Ilya Sutskever. "Exploiting similarities among languages for machine translation." *arXiv preprint arXiv:1309.4168* (2013).
14. Levy, Omer, et al. "A Strong Baseline for Learning Cross-Lingual Word Embeddings from Sentence Alignments." *arXiv preprint arXiv:1608.05426* (2016).
15. Potthast, M.; Barrón Cedeño, LA.; Stein, B.; Rosso, P. (2011). Cross-Language Plagiarism Detection. Language Resources and Evaluation. 45(1):45-62. doi:10.1007/s10579-009-9114-z.
16. Barrón-Cedeno, Alberto, et al. "On Cross-lingual Plagiarism Analysis using a Statistical Model." *PAN*. 2008.
17. Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." *Advances in neural information processing systems*. 2013.
18. Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013).
19. Le, Quoc, and Tomas Mikolov. "Distributed representations of sentences and documents." *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 2014.