1 INTRODUCTION

Word embeddings (also called vector representations or distributed representations) are high dimensional vectors consisting of real numbers which capture the syntactic and semantic meanings of the words. Closely related words are mapped near each other in a n-dimensional vector space. In other words, the cosine similarity of these vectors is close to 1. These representations aid in various natural language processing tasks like sentiment analysis, text summarization, named entity recognition etc.

2 LITERATURE SURVEY

Word embeddings have been used in a wide array of applications such as text classification, sentiment analysis and ontology enrichment [7, 19, 23]. Word embeddings have also been applied to Automatic Query Expansion [24], Biomedical Named Entity Recognition task [5] and image captioning [27]. Word embeddings have also been of use in solving problems across different languages such as Bengali [1] and Arabic [3].

The word embeddings obtained by these techniques have many applications. A paper [26] describes how training a model on material science literature is able to capture concepts like the structure-property relationships in materials. It can also suggest materials for applications well before their discovery. In another paper [8] demonstrates that the embeddings obtained from a large EHR database have yielded many improvements on disease phenotyping compared to traditional approaches. The embeddings also aided in discovering new disease-gene association. This proved that such techniques can help in minimizing costs of lab studies.Another significant application of word embeddings are recommendation systems. Many large companies use them to offer suggestions to users. For example, [9] explains how Airbnb used Word2Vec to enhance its recommendations, which helped improve its revenue stream.

Word2Vec [18] was developed by Tomas Mikolov at Google in 2013 and is one of the most popular techniques to learn word embeddings. It used local context around a word to learn embeddings and contains two variants: Skip-gram and Continuous Bag Of Words. GloVe [20] was later introduced by Stanford which used local statistics as well as global statistics by using a global word co-occurrence matrix to generate embeddings. fastText [12] was developed and introduced by Facebook in 2015 for efficient learning of word representations. Apart from the word itself, each word in fastTest is represented as a bag of character n-grams which helps preserve meaning of short words and also helps fastText tackle rare words. fastText has also helped reduce training and testing time complexities for classifiers from linear to logarithmic by making use of a hierarchical classifier based on a Huffman coding tree.

ELMo [21] and BERT [6] are two recent additions to the NLP world that achieved state of the art results over a number of NLP tasks. ELMo was developed in 2018 by AllenNLP and BERT later that year by Googles Research Group. Both differ from static models such as Word2Vec , GloVe and fastText as the embeddings they produce are contextual in nature. Unlike static models that assign a fixed embedding to a word, irrespective of the different contexts in which it may have occurred, contextual models determine embeddings based on context, hence a single word can have more than one embedding assigned to it.

There is some work done with word embeddings in the Urdu language. [10] trained the skip-gram

variant of Word2Vec with different parameters and evaluated the models over a word similarity task. [2] used Word2Vec embeddings for a Roman-Urdu to Urdu transliteration system. [22] describes how Word2Vec's embeddings were used to improve word alignments to translate 15 low resource languages (including Urdu) into English. The results exhibited improving BLEU scores. In [4], Word2Vec was used to bridge the differences between Hindi and Urdu languages. In [25], word embeddings were used to design a model for text summarization. [14] describes how word embeddings were used to enhance Named Entity Recognition approaches for the Urdu language.

Not much work has been done with regards to word-embeddings in Roman Urdu. [17] used neural word embeddings from models such as Word2Vec, fastText and GloVe in Roman Urdu Sentiment Analysis and evaluated these models over metrics such as accuracy, precision, recall and F1 score. [13] used Word2Vec clustering in their experiments for lexical normalization of Roman Urdu corpora.

2.1 Research Gap

Following the release of word2vec by Google in 2013, a lot of work has been done in the domain of word embeddings. Many more methods have been released which use different techniques to obtain these distributed representations. word2vec and some of its extensions like GloVe, fastText, doc2vec and topic2vec generate context independent embeddings (i.e. a single embedding irrespective of how many different contexts a word may be used in). Others such as BERT and ELMo can be used to obtain context dependent embeddings. These models have been trained on a number of high-resource languages like English, Spanish and French. Various state-of-the-art benchmarks have been used to evaluate the models through tasks like analogy, similarity and categorization.

Although the amount of work being done in this domain is quite extensive, one area that remains neglected is work on low-resource languages like Urdu and Roman Urdu. Due to a lack of extensive, publicly available corpora, performing such work on these languages becomes very difficult. As we conducted our literature survey on this domain, we found no work done for Roman Urdu, whereas just a single paper was published for Urdu by Samar Haider [10]. In this paper, he trained the Skip-Gram variant of the word2vec model on corpora totalling 140M+ words. He experimented with two of the parameters - context window size and vector dimensionality. To evaluate the performance of his models, he used the WordSim-353 [] and SimLex-999 [11] datasets. These datasets consist of word pairs which have been manually assigned a score based on how related or similar they are. To compare the predictions of his models, he used Spearmans Correlation coefficient.However, Haiders work was limited to just the Skip-Gram variant of word2vec. No work has been performed on any other variant or word-embedding technique for Urdu and Roman Urdu in particular. This is the research gap that we aim to close with this project. We plan to train models for both Urdu and Roman Urdu languages and the methods we will be using include word2vec (both CBOW and Skip-Gram variants), GloVe, fastText, ELMo and BERT. We will then perform a comparative analysis of these models and observe their performance on these languages. We hope that our work will pave the way for further work to be done in the field of Natural Language Processing on Urdu and Roman Urdu languages.

3 MODELS' ARCHITECTURE DETAILS

3.1 Word2Vec

Word2Vec [18] introduced two new model architectures that aimed to reduce computational complexity which was caused by the non-linear hidden layer in the Neural Network Language Models.

It consists of two models: Continuous Bag of Words and Skip-gram. In CBOW, the non-linear hidden layer is removed and the projection layer is shared for all words, which means all words are projected onto the same position. A log-linear classifier is used with a certain number of words before and after the target word which are used as the context to print the target word. In other words, the current word is predicted based on the context. Skip-gram on the other hand works by maximizing classification of a word based on another word in the same sentence. More specifically, the current word is used as an input to a log-linear classifier with a continuous projection layer to predict words within a certain range before and after the current word. In other words, the current word is used to predict the context.

3.2 Glove

GloVe [20], developed by Stanford, is a log-bilinear model that uses both global matrix factorization and local context window methods to create embeddings. It generates a word-word co-occurrence matrix over the corpus which stores counts of words occurring with each other within a given context window. Matrix factorization techniques are then applied on the matrix for dimension reduction. It trains a weighted least squares model on this matrix. The objective here is to learn vectors such that their dot product is the same as the logarithm of the words' probability of co-occurrence.

The model uses ratios of the co-occurrence probabilities rather than the raw co-occurrence probabilities as the starting point for vector learning. To handle co-occurrences that occur rarely, a weighted least squares regression model is used. This model generates two different vectors for each word. If the co-occurrence matrix is symmetric, the difference in the vectors is caused by their random initialization.

3.3 FastText

fastText [12] is a modified linear classifier which can work with a large dataset. It employs a rank constraint on the classifier, and uses a softmax function to calculate probability distributions. This hierarchical softmax, based on Huffman coding tree, helps reduce the running time from O(kh) to O(h log2 k). It further improves its efficiency by using a depth-first search algorithm, which discards small probability branches.

In addition to the word embeddings, fastText also uses a bag of n-grams to represent a word. For example, considering the word machine, with n = 3, the model would represent this word as <ma, mac, ach, chi, hin, ine, in>, where the angular brackets denote the start and the end of the word. This allows fastText to extract much more information from words than other models.

3.4 BERT

BERT differs from previous techniques by incorporating a transformer architecture that consists of multiple encoder layer stacked upon one another. It has two variants, namely: $\text{BERT}_{\text{base}}$ and $\text{BERT}_{\text{large}}$. The base model has 12 encoder layers, 768 Feed Forward Networks, 12 Attention Heads and a total of 110M parameters whereas the large model has 24 encoder layers, 1024 Feed Forward Networks, 16 Attention Heads and a total of 340M parameters.

BERT can be pretrained on a large un-labelled text corpus and can later be finetuned to any task specific dataset. Pre-training comprises of two key phases: Masked Language Modelling (MLM) and Next Sentence Prediction (NSP). In MLM 15% of token positions in the sentences passed are chosen at random and masked and the surrounding words are used to predict the masked token. BERT expects pairs of sentences concatenated together as input with a separation token between them. In NSP, sentences A and B for each pair passed are selected in a randomized manner from

the corpus such that 50% of the time B is the actual sentence that follows A and 50% of the time B it is not. The model predicts whether or not sentence B follows A or not. Once the model has been pretrained it can later be finetuned to support a variety of different NLP tasks through minor modifications to the output of the model.

3.5 ELMo

ELMo runs multi-layer forward and backward LSTMs to compute the embeddings. It starts by first assigning a character level embedding to the tokens that are passed. Character level embeddings aid in capturing morphological features of the tokens that are passed to them. These embeddings are passed through a series of layers before being forwarded to a 2-layer high-way network. The biLM model has two layers stacked together where each layer has two passes - forward pass and backward pass. The forward pass contains information about a certain word and captures the context before that word. In the backward pass, the information about the word is captured along with the context after that word. This pair of information forms the intermediate word vectors. ELMo calculates the weighted sum of the raw word vectors and the intermediate word vectors. This allows for semi-supervised learning, where the biLM is pre-trained at a large scale and easily incorporated into other neural architectures.

4 RESULTS AND ANALYSIS

4.1 Quantitative Analysis

To perform quantitative analysis, we used SimLex-999 and WordSim-353 datasets. SimLex-999 contains 999 word pairs and are used to observe how well our models capture similarity between words. WordSim-353 contains 353 word pairs which is used to measure how well it captures similarity as well as relatedness between words. Both of these datasets contain scores which are assigned by the annotators and agreed upon, that rate how similar or related the word pairs are. The following figures show what these datasets look like:

Word A	Word B	Score
kitaab	kaghaz	7.46
computer	internet	7.58
train	car	6.31

Word A	Word B	Score
		7.46
computer	internet	7.58
train	car	6.31

Since we are conducting our research in the Urdu and Roman Urdu languages, we had to translate the SimLex and WordSim datasets to Urdu and then transliterate to Roman Urdu. We translated the word pairs from English to Urdu using Googles translate service. Certain words which Google Translate failed to translate or showed irregular translations, such as single words being translated into phrases, were removed from the dataset. Words from Urdu were then transliterated to Roman Urdu using iJunoons Urdu to Roman Urdu transliteration service.

For evaluating our models on these datasets, we used Spearmans Rank Correlation which is a technique used to calculate the strength and direction between two variables. It returns a value from -1 to 1 where:

+1: a perfect positive correlation between variables
-1: a perfect negative correlation between variables
0: no correlation between variables

	WordSim-353	SimLex-999
Word2Vec CBOW	0.495	0.189
Word2Vec SG	0.485	0.186
fastText CBOW	0.406	0.165
fastText SG	0.474	0.199
GloVe	0.367	0.095
ELMo	-0.067	0.097
BERT	0.327	0.123

	WordSim-353	SimLex-999
Word2Vec CBOW	0.517	0.165
Word2Vec SG	0.527	0.205
fastText CBOW	0.465	0.122
fastText SG	0.544	0.144
GloVe	0.442	0.137
ELMo	0.156	0.079
BERT	0.039	-0.083

The Spearmans Rank Correlation is represented by the formula:

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

where,

n is the number of observations,

d_i is the difference in rank between the i^{th} observations

If we compare our Urdu scores with those of [10], we can observe that they are quite close to his own score. Whereas [10] used different embedding sizes of 100, 200 and 300, we used an embedding size of 500. [10] also varied the size of the context window and chose them to be 3, 5 and 7 while we fixed ours to be 5. [10] trained only a single model which was the Word2Vec Skip-gram model while we trained both the CBOW and Skip-gram variants of Word2Vec and on top of that trained models for fastText and GloVe as well. For the WordSim-353 dataset, the best score [10] was able to get was 0.524 on a context window size of 5 and dimensionality 200 while we were able to achieve 0.485. For the SimLex-999 dataset, [10] was able to get a best score of 0.306 on a context window size of 7 and dimensionality 300 while we got a score of 0.186. The lower score could be explained by the fact that SimLex-999 is a challenging dataset with a high variety of words. Static models such as word2vec, fastText and GloVe gave comparable results however the embeddings extracted from the contextual models such as BERT and ELMo did not score that high. This verifies the fact that raw embeddings from these models arent as useful. These models need to be fine-tuned with task-specific datasets to yield better results. The embeddings produced by these models are not fixed and change according to context.

4.2 Qualitative Analysis

To perform a qualitative analysis of the embeddings generated by the models, we essentially generated plots of two types - word-word and word-cluster. These plots show how the models project related words in a vector space. To generate these plots, we used matplotlib to create scatter plots. Since we trained the models to generate high dimensional (500) embeddings, the first thing we needed to do was transform the vectors to lower dimensions (in this case, 2). To achieve this, we

Country	Capital
افغانستان	کابل
جاپان	ٹوکیو
انگلینڈ	لنڈن
عراق	بغداد

Table 1. Urdu Country-Capital Pairs

Word	Synonym
خدا	پروردگار
خوبصورت	دلکش
ہنس	مسکرا

Table 2. Urdu Word-Synonym Pairs

used the scikit-learn library.

1. Word-Word Plots

We chose word pairs spanning various categories, like country-capital, word-synonym, word-antonym and singular-plural. The words chosen from these categories were carefully selected, in order to display meaningful plots to the viewer.

To perform dimensionality reduction for our word-word plots, we used Principal Component Analysis from the scikit-learn library. This performs linear dimensionality reduction using Singular Value Decomposition of an embedding to project it to a lower dimension. To connect the word-word pairs, we drew line segments between the plotted embeddings.

1.1 Urdu

For the plots generated for the country-capital pairs, the key thing to observe was the direction in which a country was plotted relative to its capital. The SG variant of Word2Vec does not yield consistent directions for any of the word pairs. The CBOW variant on the other hand gives the same direction for the pairs except افغانستان کابل. GloVes plot shows the pairs have two different directions. For the SG variant of fastText, the عراق بغداد pair is an outlier. The CBOW variant, unlike that of Word2Vec, does not exhibit the same consistency. The plot given by ELMos embeddings shows two different directions for two pairs of the country-capital pairs. Meanwhile, for BERT, the ٹوکیو جاپان pair is a clear outlier. For the word-synonym pairs, the main property to observe is the distance between the embeddings. The SG variant of Word2Vec plots ہنس مسکرا pair on the point while the others are at some distance. In the plot for the CBOW variant, the خوبصورت دلکش pair is the closest. The GloVe model was unable to plot any of the pairs in close proximity. The ہنس مسکرا pair is well placed in plot for the SG variant of fastText, while the خوبصورت دلکش pair is the best placed pair in the CBOW variants plot.

1.2 Roman Urdu

Country	Capital
pakistan	islamabad
japan	tokyo
england	london
iraq	bghdad

Table 3. Roman Urdu Country-Capital Pairs

Word	Synonym
khuda	parvardigaar
khoobsurat	dilkash
hans	muskura

Table 4. Roman Urdu Word-Synonym Pairs

We observed that generally the directions for the word pairs are consistent within a plot. This is true except for the iraq-bghdad pair plotted by both the variants of Word2Vec. The plots for the embeddings learned by GloVe also show the japan-tokyo and pakistan-islamabad pairs having different directions than the rest of the pairs. The SG variant of fastText shows different directions for almost every pair. The CBOW variant plots all but the iraq-bghdad pair in almost the same direction. The ELMo models plot only has the london-england and kabul-afghanistan pairs in the same direction. Meanwhile the plot from BERTs embeddings was very inconsistent.

The SG variant of Word2Vec plots the hans-muskura pair in close proximity. In comparison, the CBOW variant plots. The plots showing GloVes representations of the word, only the dilkash-khoobsurat pair has a small distance. For the SG variant of fastText, the hans-muskura pair has the best learnt representation, as they have a very small distance in between. In the CBOW variant, the dilkash-khoobsurat pair is the best represented one.

2. Word-Cluster Plots

For cluster plots, we hand-picked a list of words and used clusters of words considered the closest to these by each of the models to make the scatter plots and see how close these related words are projected in a 2D vector space.

Here, dimensionality reduction is performed using t-distributed Stochastic Neighbor Embedding (TSNE) from scikit-learn. This changes similarities between the data points to joint probabilities and attempts to reduce the Kullback-Leibler divergence between the joint probabilities of the low dimensional representations and the high dimensional data.

2.1 Urdu

The clusters plotted from the Word2Vec SG variant are not very useful. They are not as closely projected as they should be. The CBOW variants plot shows a very weird projection - two words, and , are considered similar to some word but they are plotted quite far from the rest of the words. But the clusters again are not very meaningful. GloVe is able to project slightly better clusters,

No.	Word
1.	موسم
2.	کھانا
3.	مذہب
4.	صبح
5.	باپ
6.	حکومت
7.	امریکہ
8.	پاکستان
9.	مسلمان
10.	کرکٹ
11.	مذہب
12.	محمد
13.	شہر
14.	مالک
15.	اللہ

Table 5. Urdu Words

with some being closely projected. fastTexts plots are much better, with many clusters appropriately placed. The CBOW variant shows the best projections of the clusters. Neither ELMo nor BERT were able to give a meaningful cluster, although one thing to be noticed about ELMos clusters is that all the words have similar spellings. This can be attributed to the fact that ELMo is a character-based model.

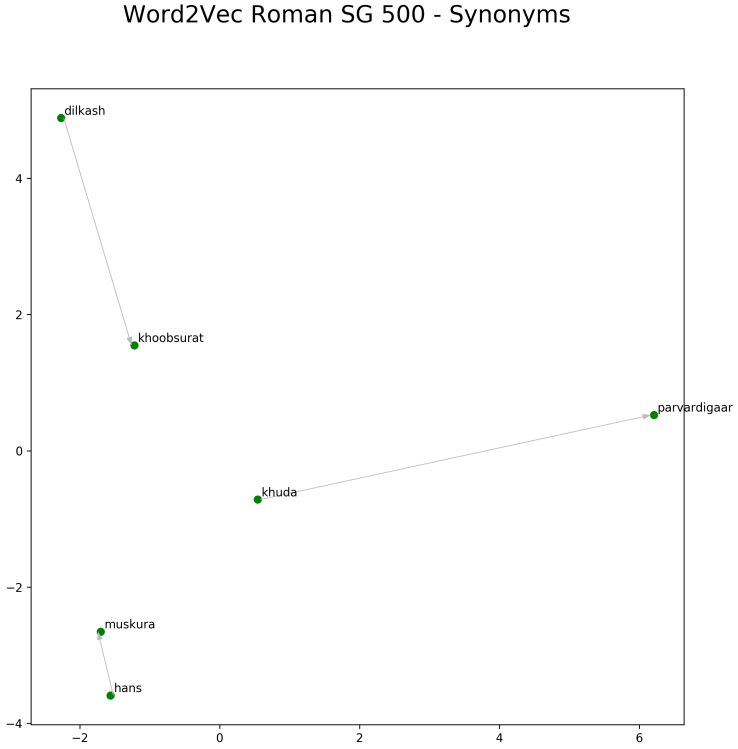
2.1 Roman Urdu

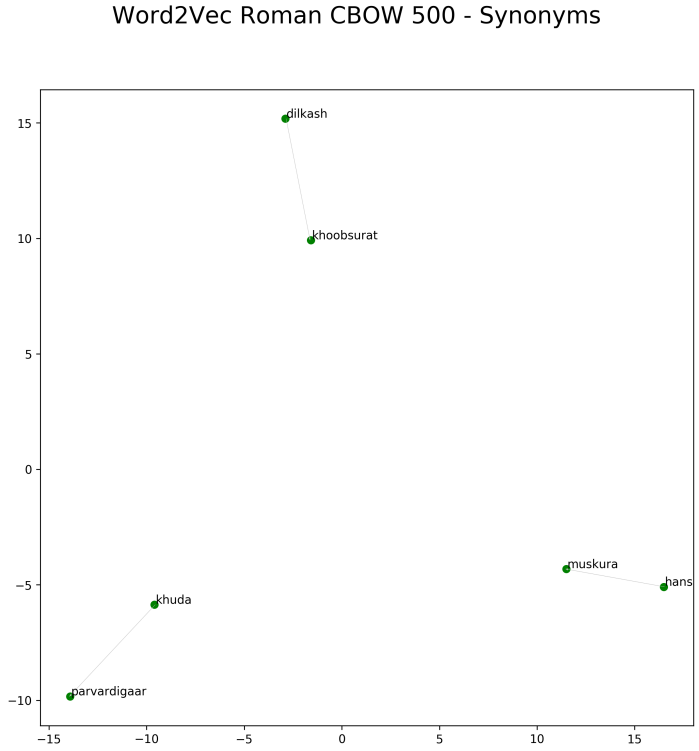
The clusters plotted by the SG variant of Word2Vec are not very meaningful - for some of the clusters, the words are closely plotted, while for others are more spread-out. The CBOW variant does much better, with most of the similar words being in close proximity. GloVes representations roughly fall in between those of the Word2Vec variants. However, fastTexts clusters were the most interesting. Due to its capture of n-gram information, the clusters it makes consists of words which have similar spellings (or in other words, these words contain a substring of the main word). For example, the cluster for the word america contains words such as american, amreeci and ameriki. Among the two variants, the CBOW variants cluster was the most closely packed. And once again, we see that ELMo and BERTs clusters are not meaningful, with the formers clusters consisting of similarly spelt words.

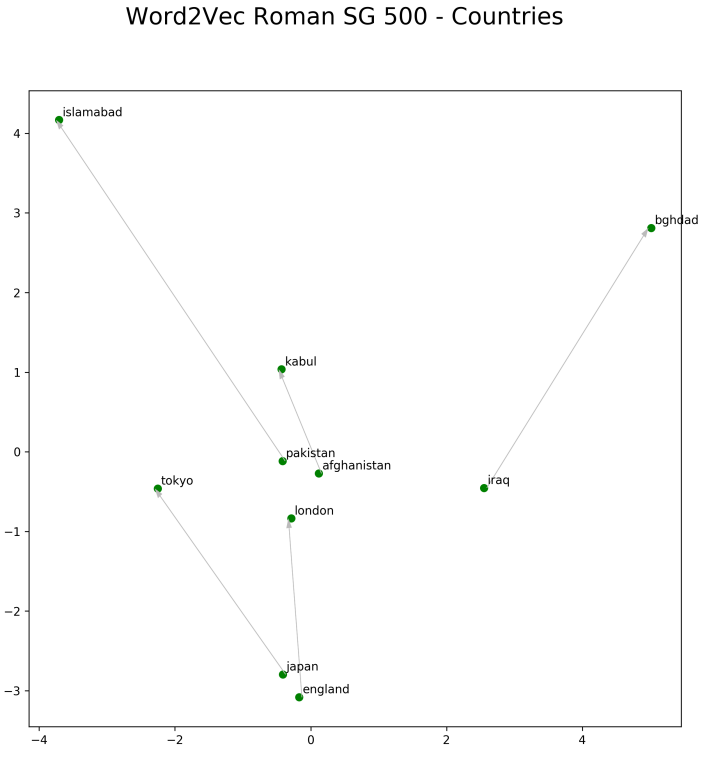
Graphs Used are given below:

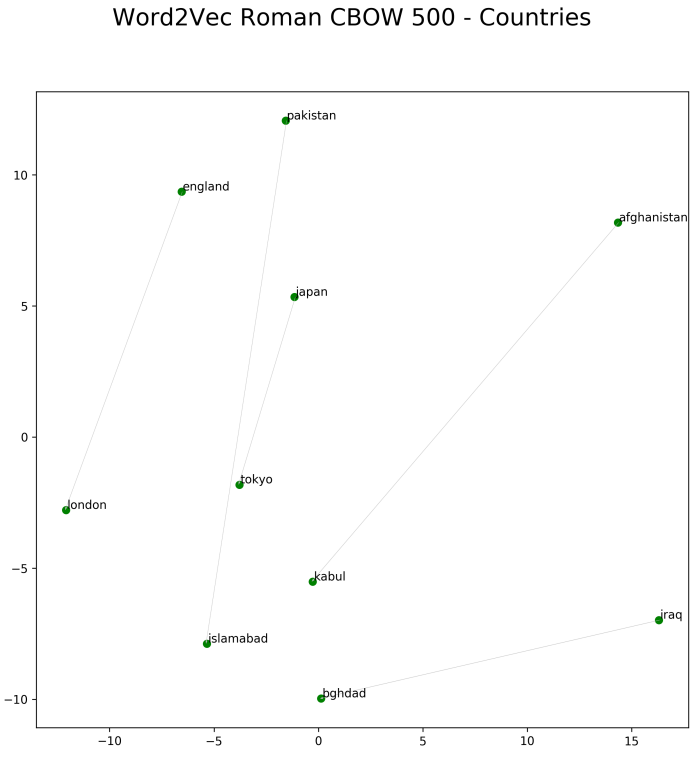
No.	Word
1.	mausam
2.	khana
3.	mazhab
4.	subha
5.	baap
6.	hukoomat
7.	america
8.	pakistan
9.	muslaman
10.	cricket
11.	mazhab
12.	mohammad
13.	shehar
14.	maalik
15.	allah

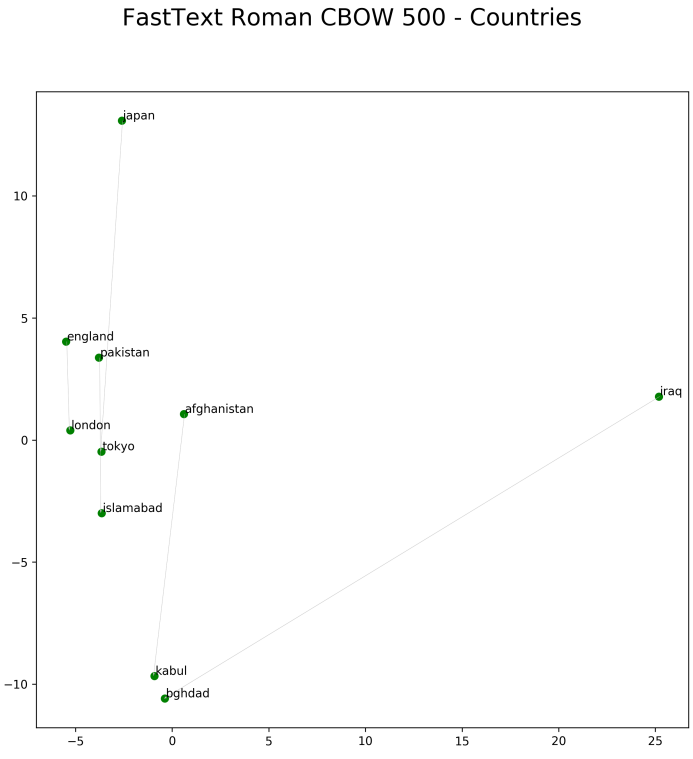
Table 6. Roman Urdu Words




Word2Vec Roman CBOW 500 - Synonyms
dilkash
khoobsurat
muskura
hans
khuda
parvardigaar
15
10
5
0
−5
−10
−15
−10
−5
0
5
10
15


Word2Vec Roman SG 500 - Countries
islamabad
bghdad
kabul
pakistan
afghanistan
tokyo
iraq
london
japan
england
4
3
2
1
0
−1
−2
−3
−4
−2
0
2
4


Word2Vec Roman CBOW 500 - Countries
pakistan
england
afghanistan
japan
tokyo
london
kabul
iraq
islamabad
bghdad
10
5
0
−5
−10
−10
−5
0
5
10
15



5 CONCLUSIONS AND FUTURE WORK

We have successfully trained models of the Word2Vec, GloVe, fastText, ELMo and BERT techniques on our Urdu and Roman Urdu datasets and performed both qualitative and quantitative analyses. The main challenge we faced was the lack of publicly available, quality datasets for both the languages. These word embedding models can be evaluated using various NLP tasks like sentiment analysis, named entity recognition and part-of-speech classification, to name a few. However, each of these techniques requires the availability of a dataset designed for that specific task, in order to fine-tune a model for it. We could not find any such datasets for these languages. Due to this reason, we resorted to translating a couple of benchmark datasets namely: Wordsim-353 and Simlex-999 for a word similarity task to evaluate our models. This turned out to be problematic in our research when we attempted to use the same for evaluating BERT and ELMo, since they require a context to be provided to give meaningful word representations. Most evaluation methods and benchmarks described in [6] and [21] are exclusive to only a few high-resource languages and translating them for the sake of analysis was outside the scope of our project. We are able to find only one compatible benchmark, XNLI, for which we evaluated our BERT-Urdu model.

To advance research in the field of NLP for Urdu and Roman-Urdu, the biggest necessity is for quality datasets designed to perform various tasks as stated earlier. For future work, a more diverse and comprehensive dataset for Urdu and Roman Urdu languages could be developed for improved analysis and comparison. Popular benchmark tests could be translated or new datasets may be developed to enable further research into this area. Newer developed models such as ALBERT [15]

and RoBERTa [16] could also be evaluated in the future and analyzed using the same methodologies that we applied in this research.

REFERENCES

[1] Adnan Ahmad and Mohammad Ruhul Amin. 2016. Bengali word embeddings and it's application in solving document classification problem. In *2016 19th International Conference on Computer and Information Technology (ICCIT)*. IEEE, 425–430.

[2] Mehreen Alam and Sibt ul Hussain. 2017. Sequence to sequence networks for Roman-Urdu to Urdu transliteration. In *2017 International Multi-topic Conference (INMIC)*. IEEE, 1–7.

[3] A. A. Altowayan and L. Tao. 2016. Word embeddings for Arabic sentiment analysis. In *2016 IEEE International Conference on Big Data (Big Data)*. 3820–3825.

[4] Riyaz Ahmad Bhat, Irshad Bhat, Naman Jain, and Dipti Misra Sharma. 2016. A house united: Bridging the script and lexical barrier between hindi and urdu. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 397–408.

[5] FX Chang, J Guo, WR Xu, and S Relly Chung. 2015. Application of Word Embeddings in Biomedical Named Entity Recognition Tasks. *Journal of Digital Information Management* 13, 5 (2015).

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:cs.CL/1810.04805

[7] Maria Giatsoglou, Manolis Vozalis, Kostas Diamantaras, Athena Vakali, George Sarigiannidis, and Kostas Chatzisavvas. 2017. Sentiment Analysis Leveraging Emotions and Word Embeddings. *Expert Systems with Applications* 69 (03 2017), 214–224. https://doi.org/10.1016/j.eswa.2016.10.043

[8] Djordje Gligorijevic, Jelena Stojanovic, Nemanja Djuric, Vladan Radosavljevic, Mihajlo Grbovic, Rob J Kulathinal, and Zoran Obradovic. 2016. Large-scale discovery of disease-disease and disease-gene associations. *Scientific reports* 6, 1 (2016), 1–12.

[9] Mihajlo Grbovic and Haibin Cheng. 2018. Real-time personalization using embeddings for search ranking at airbnb. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 311–320.

[10] Samar Haider. 2018. Urdu Word Embeddings. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

[11] Felix Hill, Roi Reichart, and Anna Korhonen. 2014. SimLex-999: Evaluating Semantic Models with (Genuine) Similarity Estimation. arXiv:cs.CL/1408.3456

[12] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of Tricks for Efficient Text Classification. arXiv:cs.CL/1607.01759

[13] Abdul Rafae Khan, Asim Karim, Hassan Sajjad, Faisal Kamiran, and Jia Xu. 2020. A Clustering Framework for Lexical Normalization of Roman Urdu. arXiv:cs.CL/2004.00088

[14] Wahab Khan, Ali Daud, Fahd Alotaibi, Naif Aljohani, and Sachi Arafat. 2019. Deep recurrent neural networks with word embeddings for Urdu named entity recognition. *ETRI Journal* (2019).

[15] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. arXiv:cs.CL/1909.11942

[16] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:cs.CL/1907.11692

[17] Faiza Memood, Muhammad Usman Ghani, Muhammad Ali Ibrahim, Rehab Shehzadi, and Muhammad Nabeel Asim. 2020. A Precisely Xtreme-Multi Channel Hybrid Approach For Roman Urdu Sentiment Analysis. arXiv:cs.CL/2003.05443

[18] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. arXiv:cs.CL/1310.4546

[19] Izzet Pembeci. 2016. Using Word Embeddings for Ontology Enrichment. *International Journal of Intelligent Systems and Applications in Engineering* 4 (2016), 49–56.

[20] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[21] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. arXiv:cs.CL/1802.05365

[22] Nima Pourdamghani, Marjan Ghazvininejad, and Kevin Knight. 2018. Using word vectors to improve word alignments for low resource machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. 524–528.

[23] Adithya Rao and Nemanja Spasojevic. 2016. Actionable and Political Text Classification using Word Embeddings and LSTM. *CoRR* abs/1607.02501 (2016). arXiv:1607.02501 http://arxiv.org/abs/1607.02501

[24] Dwaipayan Roy, Debjyoti Paul, Mandar Mitra, and Utpal Garain. 2016. Using Word Embeddings for Automatic Query Expansion. *CoRR* abs/1606.07608 (2016). arXiv:1606.07608 http://arxiv.org/abs/1606.07608

[25] Maida Shahid, Summra Saleem, Aniqa Dilawari, and Usman Ghani Khan. 2019. A Rewriter Model for Urdu Document Concision with Neural Word Embeddings. *Urdu News Headline, Text Classification by Using Different Machine Learning Algorithms* (2019), 39.

[26] Vahe Tshitoyan, John Dagdelen, Leigh Weston, Alexander Dunn, Ziqin Rong, Olga Kononova, Kristin A Persson, Gerbrand Ceder, and Anubhav Jain. 2019. Unsupervised word embeddings capture latent knowledge from materials science literature. *Nature* 571, 7763 (2019), 95–98.

[27] Cheng Wang, Haojin Yang, Christian Bartz, and Christoph Meinel. 2016. Image Captioning with Deep Bidirectional LSTMs. In *Proceedings of the 24th ACM International Conference on Multimedia (MM 16)*. Association for Computing Machinery, New York, NY, USA, 988997. https://doi.org/10.1145/2964284.2964299