Character Word Embedding for NLP tasks in Indian languages

Anirban Majumder[1] and Amit Kumar[2] and

[1,2]Indian Institute of Technology, Kanpur
[1]manirban@cse.iitk.ac.in,[2]amitkmr@cse.iitk.ac.in

Abstract

In the recent time Word Embeddings have been used as unsupervised approach to achieve results comparable to that of supervised methods which use handcrafted features.But information about word morphology and shape is normally ignored when learning word representations.Character level embedding can capture the intra-word information specially when dealing with morphologically rich languages. Here we propose to use neural network that learns character level representation of words and associate them with usual word representations to perform morphologically rich text such as POS tagging. Our character-level word embedding essentially takes word's morphological segmentation into account (affixes,root etc.) and generate the embedding by giving preferences to the various parts of the segmentation. In the result of the word similarity tasks also it produces good result. The method is not language specific, so it can be implemented for other languages our given code.

Introduction

Language modeling is a fundamental task in artificial intelligence and natural language processing (NLP), with applications in speech recognition, summarization, and machine translation. A language model is formalized as a probability distribution over a sequence of strings (words), and traditional methods usually involve making an n-th order Markov assumption and estimating n-gram probabilities via counting and subsequent smoothing. Now a new trend has come when the approach for unsupervised learning of words came into the frontier. Approaches using word embeddings have really shown results comparable with the traditional supervised methods and it has opened a new field in natural language processing.

Word embeddings are based on the idea that contextual information alone constitutes a viable representation of linguistic items, in stark contrast to formal linguistics and the Chomsky tradition. Recently Neural language model has shown some improvement over the traditional semantic models. Neural Language Models (NLM) address the data sparsity issue through parametrization of words as vectors (word embeddings) and using them as inputs to a neural network Bengio et al. 2005 [5]; Mikolov et al. 2013 [4].

Related work

Work on learning of representations for NLP has focused exclusively on the word level. However, information about word morphology is very crucial for various NLP tasks which has rich syntactic requirements. For that, one should also take into account the characters of the words which encaptures the various syntactic features of the word. Usually, when a task needs morphological or word shape information, the knowledge is included as handcrafted features (Collobert,2011) [1]. Recently a paper by Santos et al. 2014 [6] has taken a convolutional approach to obtain the character-level word embedding from the individual character embedding using CNN and showed that the results for tasks such as POS tagging has improved.

Here we propose a more general approach to obtain a character level word embedding using morphological segmentation for words. We implement the work by Dasgupta and Ng [2] to get the word's segmentations for generating the word's embedding from it's characters. The word similarity results that we obtained from our experiment supports the utility of our approach.

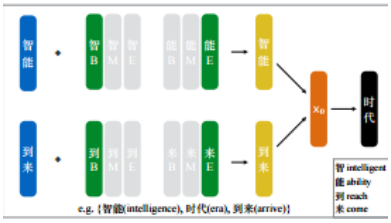

Figure 1: Position-based character embeddings.

Proposed Model

We are using CWE embedding by Chen et al. [8] to obtain the character level embedding for the words. Simple modifications have to be done for different languages. After that we obtain the morphological segmentation of words and apply weights to different word segmentation using a basic rule. After that we obtain the word level embedding using

the character embeddings. For the experimental result, we evaluate our character level word embedding by generating word similarities in 3 languages, English,Hindi and Bengali. We also evaluate the model by Santos et al. by applying LSTM for word prediction tasks.

Character Embedding

The procedure for character embedding is same as that of word embedding approach of CBOW, Word2Vec. We followed the approach of Chen et al. [8]. In the input for the CBOW, we feed both the word embedding and the character embedding of the current word. When the error is back-propagated through the neural network, it updates both the word and the character embedding matrix.

Char-level word embedding

In CWE embedding by Chen et al., to obtain the character embedding we use the position based approach they have shown. The proposed approach shown in Fig.1 is almost similar like word2vec CBOW approach. In place of word embeddings in the input of the neural network,it just give combination of character and word embeddings. For a letter it gives 3 embeddings based on its position(Begin,Middle,End).

colorless	Financially	Electricity
colourless	Functionally	Eccentricity
cordless	Semantically	Intercity
careless	Originally	Elasticity
countless	Fantastically	Plasticity
clueless	Traditionally	Apostolicity

Table 1: Similar words using Character Level Embedding

Word morphology analysis

For word morphological segmentation, we use the algorithm implemented by the Dasgupta et al. in his paper [2]. To get the word's affixes, it calculates a word-root frequency and use that to split the words into prefixes+root+suffixes. It also gives as output the top prefixes and suffixes which can be used for ranking them when giving them the weights to calculate the character level embedding of words.

Weight Distribution to the word segments

After parsing the words into segments, the words are weighted by a very basic rule. From the root part, we assign incremental weights in both the sides i.e. for the suffixes and prefixes. It is a very basic assumption, but seems to have a good result for the word syntactic similarity tasks. After getting the embeddings for the words, we divide the embedding by the length of the word.

$$x_j = \frac{1}{N_j} \sum_{i=1}^{n} c_k * w_k$$

Datasets

For 3 languages, we use the WIKI corpus. We use the WIKI extracted corpus collected by Rami Ali Rfou(Polyglot). For the bengali corpus, we cleaned the corpus using a WIKI extractor [7] and then use the unicode range for bengali to clean other language letters, present in the corpus. Wikipedia english corpus Specs : (16 million words, Vocab Size: 70k),Wikipedia hindi corpus specs : (14 Millions Word, Vocab Size: 75004), Wikipedia Bengali corpus (After cleaning) specs: (5 million words, Voacb size: 59600)

Experimental results

We have 2 models on which we have evaluate our results. The first one is the weighted approach and the other one is the CNN approach by Santos. We apply the models for prediction in word prediction in English,Hindi and Bengali. So on our model,we run the word similarity tasks which is in the Table 1 and in the Figure 2. From those we can see that word similarity result is quite good. For the Santos approach, we uses the LSTM-CNN approach for word prediction tasks and calculate its perplexity for different languages.

भारतीय	प्राप्त	प्रयोग
भरतीय	पर्याप्त	प्रोग
जातीय	समाप्त	प्रियोग
उत्तरीय	व्याप्त	पुनर्प्रयोग
काकतीय	पुनःप्राप्त	स्त्रीरोग
कार्तीय	व्याप्त	नाट्यप्रयोग

Figure 2: Similar Word (5-NN) Results for Hindi

আলোচনাকালে	আমদানিকে	সরলভাবে
শুনানিকালে	দোকানদারকে	মিলিতভাবে
বর্তমানকালে	অন্যদিকে	দলনিরপেক্ষভাবে
খননকালে	আফ্রিকানকে	জাতীয়ভাবে
অপহরণকালে	আরেকজনকে	অবিকৃতভাবে
নির্ভরশীলতাকে	মহাদলকে	বাড়াবে

Figure 3: Similar Word (5-NN) Results for Bengali

The perplexity is shown in the Table 2. The perplexity for hindi shown is only for 5 epochs for the shortcomings of our processor speed. So it's not giving any comparable results yet. We hope that on GPU it will give a better result as shown in the paper by Kim [3]. But we can see that perplexity is quite low for the character level word embedding than the word-level embedding.

Conclusion

We have implemented these two models which show quite comparable results. Character level word embedding pre-

Perplexity	LSTM-Word	LSTM-CharCNN
English(ep=25)	97.6	92.3
Hindi (ep=5)	664.68	601.85

Table 2: Word Prediction Results

serves the intra-word structure required for the morphologically rich tasks. We have not tested the composition of our obtained character level embedding and general word level embedding. But we hope observing the found results that with a right compositional rule, the joint embedding can be of great interest for various NLP tasks.

Future Works

We have not combined our character embedding with the traditional word embedding and evaluate the combined approach for Word similarity tasks. This is we have in mind for future work. Another application can be to change the procedure of obtaining the character embedding. Here we use the conventional CBOW approach of word2vec to get the character and the word embedding. In place of this, we can use RNN or LSTM like network which will try to predict the next character and generate the character embedding.

References

[1] Ronan Collobert et al. "Natural Language Processing (almost) from Scratch". In: *CoRR* abs/1103.0398 (2011). URL: http://arxiv.org/abs/1103.0398.

[2] Sajib Dasgupta and Vincent Ng. "High-Performance, Language-Independent Morphological Segmentation". In: *NAACL HLT 2007: Proceedings of the Main Conference*. 2007, pp. 155–163.

[3] Yoon Kim et al. "Character-Aware Neural Language Models". In: *CoRR* abs/1508.06615 (2015). URL: http://arxiv.org/abs/1508.06615.

[4] Tomas Mikolov et al. "Efficient Estimation of Word Representations in Vector Space". In: *CoRR* abs/1301.3781 (2013).

[5] Frederic Morin and Yoshua Bengio. "Hierarchical probabilistic neural network language model". In: *AISTATS05*. 2005, pp. 246–252.

[6] Cicero D. Santos and Bianca Zadrozny. "Learning Character-level Representations for Part-of-Speech Tagging". In: *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. Ed. by Tony Jebara and Eric P. Xing. JMLR Workshop and Conference Proceedings, 2014, pp. 1818–1826. URL: http://jmlr.org/proceedings/papers/v32/santos14.pdf.

[7] "Wikipedia Extrator". In: (). URL: https://github.com/bwbaugh/wikipedia-extractor.

[8] Zhiyuan Liu Maosong Sun Huanbo Luan Xinxiong Chen Lei Xu. "Joint Learning of Character and Word Embeddings". In: (2015).