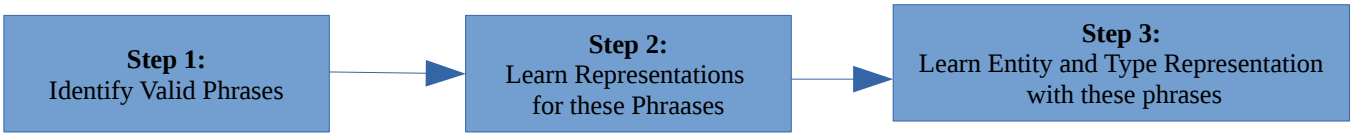
Word -> Phrase Based Approach

Till now, we were working on learning representation for Entity and Type using word-embeddings.

As per our observation sometimes words only may not able to give correct context. Phrases will be more useful like, was attacked, accused in, attacked at etc.

Examples:



Step 1: Identify valid phrases

Phrase could be learnt or identified using following methods:

i) Bag of word-pair: Generate all the bigram pair and learn their representation.

Example:

Ram was going to Agra.

to

Ram_was was_going going_to to_Agra

- This approach will generate the noises.

ii) Bag of word-pair with Seed Phrases

Learn all word pair and get the pairs which are close to ‘Accused_of’, ‘was_attacked’ etc.

iii) Collocations-based-phrases:

Collocations: Words that appear frequently together, and infrequently in other contexts. [1]
Collocations based phrase were find based on the score presented in Word2Vec paper [1]

There were 2155 bigram phrases identified using collocations. Few importnt ones are:

seriously_injured, been_arrested, were_killed,

iv) Verb-phrases

This is another way of finding these phrases. Like was_attcked, were_killed etc. But it will not cover the phrases likely prime_accused, seriously_injured, accused_in etc.

v) Relation-phrases: These are the phrases which tasks about the relation between two entities. Using OpenIE these phrase could be learnt and will cover verb-phrase.

Step 2: Learn Representation for these Phrases:

Using idea of Word2Vec these phrases could be replaced with the tokens in the corpus and their phrase could be learned.

Using Pretrained Vectors: In case of word pair, these vector of these words could be concatenated and for all cases we can take average of vector of member words.

Step 3: Learn representation for Entity and Type

By replacing these phrases by single token, we can model them using earlier word based approaches.

Reference

1. Word2Vec paper: Distributed Representations of Words and Phrases and Their Compositionality. Mikolov, Tomas and Sutskever, Ilya and Chen, Kai and Corrado, Greg and Dean, Jeffrey, Proceedings of the 26th International Conference on Neural Information Processing Systems 2013

4 Learning Phrases
As discussed earlier, many phrases have a meaning that is not a simple composition of the meanings of its individual words. To learn vector representation for phrases, we first find words that appear frequently together, and infrequently in other contexts.[Collocations]

For example, “New York Times” and “Toronto Maple Leafs” are replaced by unique tokens in the training data, while a bigram “this is” will remain unchanged.

This way, we can form many reasonable phrases without greatly increasing the size of the vocabulary; in theory, we can train the Skip-gram model using all n-grams, but that would be too memory intensive. Many techniques have been previously developed to identify phrases in the text; however, it is out of scope of our work to compare them. We decided to use a simple data-driven approach, where phrases are formed based on the unigram and bigram counts, using

$$\text{score}(w_i, w_j) = \frac{\text{count}(w_i w_j) - \delta}{\text{count}(w_i) \times \text{count}(w_j)}.$$

The δ is used as a discounting coefficient and prevents too many phrases consisting of very infrequent words to be formed. The bigrams with score above the chosen threshold are then used as phrases.