Details of weight computation in *LPBridge* algorithm

Quadruples on the shortest path in the bridge representation of a sentence form the features. Words, links and thereby the quadruples in a bridge need to be weighed differently based on their importance in connectivity extraction. Consider the sentence, *“All nerves including those innervating the BR1, project to the BR2 and are somatotopically organized.”*. The bridge for this sentence is,
*[[‘Os’, ‘innerv’, ‘BR1’, 1], [‘Mg’, ‘those’, ‘innerv’], [‘Op’, ‘includ’, ‘those’, 1], [‘MX*p’, ‘n’, ‘includ’, 1], [‘Sp’, ‘n’, ‘and’, 2], [‘VJlpi’, ‘project’, ‘and’, 2], [‘MVp’, ‘project’, ‘to’, 2], [‘Js’, ‘to’, ‘BR2’, 2]]* , where the noun *‘nerves’* is denoted as *‘n’*.
The weight of a quadruple *Q=<Link(l), Left word(lw), Right word(rw), Context(c)>* is based on the weights of the words and links making it up. Weight of a word or link *e* is based on the importance of *e* in the particular context *c* and is calculated using Eq. 1. The discrimination ability of *e* with respect to classifying the connection between the brain regions is used to calculate the importance of *e*. This can be looked upon as a kind of entropy measuring the discernibility. Higher the discrimination ability, lower the entropy. Frequency of occurrence is used to scale the entropy to arrive upon the weight of the link/word *e*. Thus, weights of all links and words in the training data are computed for each of the contexts.

$$\begin{aligned} weight(e) &= log\left[Freq(e) * \frac{1}{Ent(e)}\right] \\ Freq(e) &= log(frequency(e)) \\ Ent(e) &= -PlogP - NlogN \end{aligned} \tag{1}$$

where *P* and *N* are the ratio of positive sentences and negative sentences with *e* respectively, computed over context *c* in the training corpus and *frequency(e)* is the number of times *e* has occurred in the corresponding context *c*.
Weight of the quadruple $Q = < l, lw, rw, c >$ is calculated as in Eq.2.

$$weight(Q) = \frac{weight(l) + weight(lw) + weight(rw) + weight(c)}{4} \tag{2}$$

where weights for *l*, *lw* and *rw* are calculated using Eq.1 and the choice of weights for *c* is explained in the next section.
The final Bridge representation for the sentence will be,
*[[‘Os’, ‘innerv’, ‘BR1’, 1], **0.37**], [[‘Mg’, ‘those’, ‘innerv’, 1], **0.44**], [[‘Op’, ‘includ’, ‘those’, 1], **0.59**], [[‘MX*p’, ‘n’, ‘includ’, 1], **0.25**], [[‘Sp’, ‘n’, ‘and’, 2], **0.43**], [[‘VJlpi’, ‘project’, ‘and’, 2], **0.66**], [[‘MVp’, ‘project’, ‘to’, 2], **0.68**], [[‘Js’, ‘to’, ‘BR2’, 2], **0.44**]].*
It can be noticed that informative quadruples like *[’VJlpi’, ’project’, ’and’, 2]* and *[’MVp’, ’project’, ’to’, 2]* which are important connectivity indicators have higher weights compared to quadruples *[’MX*p’, ’n’, ’includ’, 1]* and *[’Os’, ’innerv’, ’BR1’, 1].*

Experimental procedure and parameter choices

In the training phase, each sentence in the training set is preprocessed as described in section *Preprocessing* and represented in the corresponding feature representation. In the surface level representation, each sentence is represented as a vector of words. Similarity between the two sentences is computed as a weighted dot product between their vector representations. Weights (w_l,w_m,w_r) assigned to the left, middle and right vectors were set at ($w_l = 0.3, w_m = 0.6, w_r = 0.1$). The intuition behind the values is that domain experts found that the middle context contained most indicators of connectivity followed by left and then the right context. The same intuition for the middle context has been adopted by Agichtein E. *et al.*, 2000 also. Similar vectors are clustered together and each cluster center, computed as the mean of all vectors in the cluster, forms a pattern. In the link parse representation, each sentence in the training set is parsed by the link parser and represented in the form of its Bridge using the Iterative Least Cost Parsing approach. The weights associated with the words and links in the bridge representation are calculated to derive the weight of quadruples making up the bridges. In this weight computation, *weight(c)* for left, middle and right contexts are set at 0.3, 0.6 and 0.1 respectively with the same domain intuition about the importance of contexts. Pattern generation from the training set bridges is achieved by grouping similar bridges together using the weighted edit distance measure. The procedure *Substitute* in the Weighted Bridge Edit Distance algorithm computes the penalty values based on different similarity checks between a quadruple pair, where the corresponding left word, right word and link in the quadruple pair are matched as shown in Table 1 .

Penalty value	Condition
0.3	One word different
0	Links and corresponding left and right words match
0.3	Links match, but only one of the corresponding words match
0.6	Links match, but none of the corresponding words match
1	Links do not match, but only one of the corresponding words match
2	Neither links, nor, any of the corresponding words match

Table 1. Substitute penalty values used in Weighted Bridge Edit Distance algorithm

The generated patterns are ranked by their confidence score and the highly confident patterns form the pattern bank, representative of the connectivity statements in the training corpus. In the testing phase, a test sentence is classified as containing a positive connection if its confidence is more than the threshold τ_c. The confidence value is a function of the similarity of test sentence and the most similar, top *k* confident patterns generated during the training phase.
10 fold cross validation has been performed on the training data to optimize the parameters used by the algorithm. In our experiments, values of the parameters were chosen by cross validation. In the *mulBRWhiteText* dataset, for the connectivity word features, the similarity threshold for clustering $\tau_{sim} = 0.11$ and confidence threshold $\tau_c = 0.22$ was arrived upon by cross validation. Similarly, in case of link parse representation, value of parameters *k=2* and $\tau_c = 0.4$ were derived using 10 fold cross validation. The remaining parameters used by the algorithm, though set to specific values, the user of the system can configure these values to suit the needs of expected precision and recall.
The models trained on the *mulBRWhiteText* dataset are applied with the corresponding parameter setting to the *10NeuroPubMed* dataset and results are reported.

References

Agichtein, E., Gravano, L. (2000, June). Snowball: Extracting relations from large plain-text collections. In *Proceedings of the fifth ACM conference on Digital libraries* (pp. 85-94). ACM.