International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 04 Issue: 05 | May -2017 www.irjet.net p-ISSN: 2395-0072

# Computing textual semantic similarity for short texts using different similarity measures

**Riya A Gandhi[1], Dr Vimalkumar B Vaghela[2]**

[1]*Student, Dept. Of Computer Engineering, L D college of engineering, Gujarat, India*
[2]*Assistant Professor, Dept. Of Computer Engineering, L D college of Engineering, Gujarat, India*

---

**Abstract -** *Semantic similarity of short text is the method of natural language processing which is widely used in natural language processing, opinion mining, text mining, text summarization, information retrieval and recognizing textual entailment(RTE), etc. Semantic similarity reflects the semantic relation between the meaning of two sentences. Sentence similarity is used to access the likelihood between phrases. This paper presents various methods which shows similarity between two sentence pairs, performance of the methods and importance of various method.*

***Key Words***: **Semantic Similarity, RTE, STS, WSD, etc.**

## 1.INTRODUCTION

The importance of sentence semantic similarity measures The sentence semantic similarity measures are important in natural language research because of increasing applications in text-related research fields. Semantic similarity methods are classified into three types –corpus based, ontology based and hybrid approach[1]. The first method calculates the similarity from syntactic information and semantic information that they contain. In this method there are three similarity functions to derive generalized text semantic similarity. In first function string similarity is calculated then in second function semantic similarity is calculated. After that there is semantic word order function is there to incorporate semantic information in this method. At last string similarity, semantic similarity and common word order similarity are combined and normalized to calculate overall text similarity and this method is called STS(Semantic Text Similarity). The ontology based method is omiotis. It is an ontology based algorithm and based on WordNet and WSD (Word-sense disambiguation). Omiotis uses various POS(part-of-speech) and semantic relations like synonymy, antonymy, hyponymy, etc. It extends Semantic Relatedness(SR) measure between the words. It is based on the semantic links between the words according to a word thesaurus which is WordNet. In Omiotis SR in word level and statistical information in the text level is integrated and gives final SR score between texts. SyMSS(syntax-based measure for short-text semantic similarity) uses WordNet measures and parse tree. SyMSS uses grammar parser to obtain the parse tree. It is the new method which considers the syntactic information and it uses this information in WSD for reducing word matching and time complexity. STATIS is the hybrid measure which combines Word Net based and corpus based word similarities. It evaluates two sentences in the form of two vectors and obtain semantic similarity between the sentences by using vector space model(VSM). In STASIS VSM semantic similarity methods and word order similarity methods are combined to compute the sentence similarity. STATIS does not perform preprocessing tasks like stop words and meaningless words removal which will result in inaccurate similarity score. Omiotis and SyMSS reduce the ambiguity between words using the syntactic information, POS and parse tree, respectively, to match words with the same syntactic role.

## 2. METHODS OF SEMANTIC SIMILARITY

### 2.1 Corpus-Based Word Similarity and String Similarity(STS)

This method measures semantic similarity of text using corpus-based measure of semantic word similarity. This method mainly focus on measurement of semantic similarity between two sentences or short paragraphs and it uses semantic and syntactic information to evaluate the similarity of two texts.

1. String Similarity between Words

For calculating string similarity, Longest common subsequence (LCS) measure is used and some modification and normalization is done for evaluation of string similarity. Three different modified versions of LCS is used and then it will take a weighted sum of these and normalized LCS. It divides the length of the longest common subsequence by the length of the longer string which is called LCSR (longest common subsequence ratio). But it does not take into consideration the length of the shorter string which sometimes have noteworthy impact on the similarity score. Therefore computing the normalized longest common subsequence (NLCS) which takes into account the length of both the shorter and the longer string which is,

$$v_1 = NLCS(r_i, s_j) = \frac{length(LCS(r_i, s_j))^2}{length(r_i) \times length(s_j)} \qquad (1)$$

While in classical LCS, the common subsequence needs not be consecutive, in database schema matching, consecutive common subsequence is important for a high degree of matching. Therefore maximal consecutive longest common subsequence which is starting at character1, *MCLCS*1 (Algorithm [2]) and maximal consecutive longest common subsequence starting at any character n, *MCLCS*n (Algorithm

[2]) is used. Then both *MCLCS*1 and *MCLCS*n are normalized and it is called it normalized *MCLCS*1 (*NMCLCS*1) and normalized MCLCSn (NMCLCSn), respectively.

$$v_2 = NMCLCS_1(r_i, s_j) = \frac{length(MCLCS_1(r_i, s_j))^2}{length(r_i) \times length(s_j)} \quad (2)$$

$$v_3 = NMCLCS_n(r_i, s_j) = \frac{length(MCLCS_n(r_i, s_j))^2}{length(r_i) \times length(s_j)} \quad (3)$$

Then taking the weighted sum of the individual values *v*1, *v*2, and *v*3 , it will determine the string similarity score, where *w*1, *w*2, *w*3 weights and *w*1+*w*2+*w*3 = 1.

Therefore, the similarity of the two strings is:

$$\alpha = w_1 v_1 + w_2 v_2 + w_3 v_3 \quad (4)$$

2. Semantic Similarity between Words

PMI-IR uses Point wise mutual information for computing similarity of words and it is based on corpus based similarity method, defined as follows:

$$\text{PMI}(w_1, w_2) = \log p(w_1 \text{AND} w_2)/p(w_1)p(w_2) \quad (5)$$

*w*1 and *w*2 are the two words. $p(w1 \text{ AND } w2)$ is the probability that the two words co-occur. $p(w1) \cdot p(w2)$ is the probability that *w1* and *w2* will co-occur, If *w*1 and *w*2 are statistically independent. If they are not independent, and they have a tendency to co-occur, then $p(w1 \text{ AND} w2) > p(w1) \cdot p(w2)$.

Second Order Co-occurrence PMI (SOC-PMI) is used for word similarity method. The main advantage of using SOC-PMI is that it can calculate the similarity between two infrequent words, because they co-occur with the same neighboring words. The method considers the words that are common in both lists and aggregate their PMI values (from the opposite list) to calculate the relative semantic similarity. The pointwise mutual information function for only those words having $f\,b(ti, w) > 0$,

$$f^{pmi}(t_i, w) = log_2 \frac{f^b(t_i, w) \times m}{f^t(t_i) \times f^t(w)} \quad (6)$$

where $f\,t\,(ti)$ indicates that how many times the type *ti* appeared in the entire corpus, $f\,b(ti, w)$ indicates how many times word *ti* appeared with word *w* in a context windows and *m* is total number of tokens in the corpus. Now, for word *w*, define a set of words, *X w*, sorted in descending order by their PMI values with *w* and taken the top most $\beta$ words having

$f\,pmi(ti, w) > 0$.

$$X^w = \{X_i^w\}, where\ i = 1,2 \ldots \beta\ and$$

$$f^{pmi}(t_1, w) \geq f^{pmi}(t_2, w) \geq \cdots \geq f^{pmi}(t_{\beta-1}, w) \geq f^{pmi}(t_\beta, w)$$

A rule of thumb is used to choose the value of $\beta$. The $\beta$-*PMI summation* function for word *w*1 with respect to word is: (7)

$$f(w_1, w_2, \beta) = \sum_{i=1}^{\beta} (f^{pmi}(X_i^{w_1}, w_2))^\gamma$$

where, $f^{pmi}(X_i^{w_1}, w_2) > 0$, which sums all the positive PMI values of words in the set *X w*2 also common to the words in the set *X w*1 . This function actually aggregates the positive PMI values of all the semantically close words of *w*2 which are also common in *w*1's list and $\gamma$ should be $\gamma > 1$. The *semantic PMI similarity* function between the two words, *w*1 and *w*2,

$$Sim(w_1, w_2) = \frac{f(w_1, w_2, \beta_1)}{\beta_1} + \frac{f(w_2, w_1, \beta_2)}{\beta_2} \quad (8)$$

3. Common Word Order Similarity between Sentences

It is an optional method. It will consider a pair of sentences, P and R having tokens m and n respectively, that is, P = p1, p2, . . . , pm and R = r1, r2, . . . , rn and n ≥ m. If number of tokens in P is greater than in R, we switch P and R. Then count the number of pi 's (δ) for which pi = r j , for all p ∈ P and for all r ∈ R. That means, there are δ tokens in P that exactly match with R, where δ ≤ m. After that remove all δ tokens from P and put them in X and from R in Y, in the same order as they appear in the sentences. That is, X = {x1, x2, . . , xδ} and Y = {y1, y2, . . . , yδ}. Then replacing X by assigning a unique index number for each token in X starting from 1 to δ, is, X = {1, 2, . . . , δ}. Based on this, also replace Y where X = Y. The equation for measuring the common-word order similarity of two texts is:

$$S_0 = 1 - \frac{|x_1 - y_1| + |x_2 - y_2| + \cdots + |x_\delta - y_\delta|}{|x_1 - x_\delta| + |x_2 - x_{\delta-1}| + \cdots + |x_\delta - x_1|} \quad (9)$$

4. Overall Sentence Similarity

In this above three similarity functions are combined to calculate the overall sentence similarity which derives the similarity score between 0 and 1 both inclusive that will indicate the similarity between two texts *P* and *R* at semantic level. The method contains 6 steps given in ref paper[2].

## 2.2 SyMSS: A syntax-based measure for short-text semantic similarity

SyMSS uses the hierarchical structure and different glosses associated with each concept to find the semantic similarity between concepts. Here there three types of measures[3] :

1. Path-based measures

In Path based measures, the length of path between two concepts is used to measure the similarity between concepts. There are two different path based measures:

Path measure [PATH][6]: It will use the path length between two concept and measure the similarity. This measure takes into account only "is-a" relations.

Hirst and St. Onge measure [HSO]: This measure will takes into account many other WordNet relations, not only the "is-a" relation (antonyms, synonyms…).

2. Information content measures

This type of measures are used to measure the specificity of a concept, which is higher for more specific concepts. There are three different measures of this type:

Resnik measure [RES][7]: If two concepts are semantically similar in terms of the amount of information they share then RES measure is used to calculate similarity. It is calculated as the information content of their lowest common subsumer in the hierarchy:

$$Sim_{res}(c_1, c_2) = IC(lcs(c_1, c_2)) \quad (10)$$

Jiang and Conrath measure [JCN][8]:This measure checks that whether the sum of the individual information contents is similar to their lowest common subsumer, if similar, then the concepts are close together in the hierarchy.

$$Sim_{jcn}(c_1, c_2) = \frac{1}{IC(c_1) + IC(c_2) - 2 * IC(lcs(c_1, c_2))} \quad (11)$$

Lin measure [LIN][9]: this computes simply the ratio of the information content of the lowest common subsumer to the information content of each of the concepts. Bellow is the equation for LIN measure:

$$Sim_{lin}(c_1, c_2) = \frac{2 * IC(lcs(c_1, c_2))}{IC(c_1) + IC(c_2)} \quad (12)$$

3. Gloss-based measures

There are glosses associated with each concept in the WordNet. Gloss based measure use this kind of information. Two types of Gloss-based measure:

Extended Gloss Overlap measure [LESK-E][10]: The measure evaluates similarity of two concepts from the overlapping of the glosses connected with each concept and with their related concepts in WordNet.

Gloss Vectors measure [VECTOR][11]: Each concept is represented as gloss vector, it finds the similarity by averaging gloss co-occurrence data and by calculating the cosine between these vectors.

Semantic similarity between sentences- The phrases that have the same syntactic function in two sentences are summed together to calculate the similarity between two sentences. The equation is:

$$sim(s_1, s_2) = \frac{1}{n}\sum_{i=1}^{n} sim(h_{1i}, h_{2i}) - l.PF \quad (13)$$

n is number of phrases in sentences s1 and h11,..h1n are heads, similar for sentence2; and the phrases of h1i and h2i have the same syntactic function. In the case, where the sentences have one syntactic roles that are only present in one of the sentences, then if one sentence has a phrase not shared by the other, a PF (penalization factor ) is introduced to reflect the fact that one of the sentences has extra information. The heads that are not present in WordNet (for example, pronouns) are ignored in the calculation unless the same word shares the same syntactic role in both sentences.

## 2.3 STASIS

In this method first the joint word set containing only distinct word from the sentences is formed dynamically as shown in the figure. After that by using lexical database, the raw semantic vector and order vectors are derived for every sentence pair. Then Semantic vectors obtain from the raw semantic vectors and word is weighted by using information content derived from a corpus. Semantic similarity and order similarity is computed. Finally, the sentence similarity is computed by combining semantic similarity and order similarity.[4]

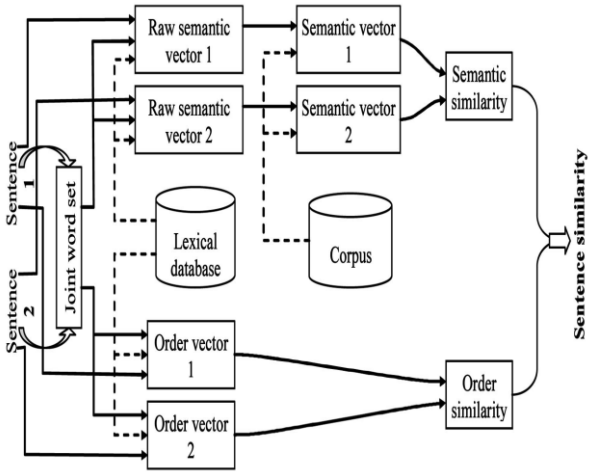


Fig. 1. Sentence similarity computation diagram.

1. Semantic Similarity Between Words

The bellow figure shows the hierarchical semantic knowledge base and depth of word. Based on the path length given in this, the Similarity between words is determined.

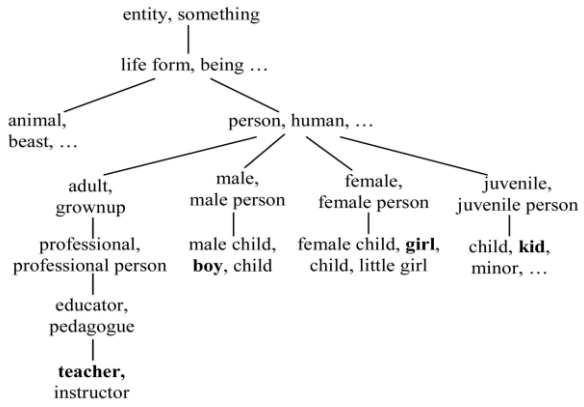


Fig. 2. Hierarchical semantic knowledge base.

$w_1$ and $w_2$ are words and similarity is calculated as the function of path length and depth :

$$s(w_1,w_2)=f(l,h), \quad (14)$$

$l$ = the shortest path length between words $w_1$ and $w_2$ , $h$= depth of subsumer in hierarchical semantic nets. This can be written as

$$s(w_1,w_2)=f_1(l).f_2(h). \quad (15)$$

$f_1$ and $f_2$ are transfer function of path length and depth, respectively and similarity ranges between [0,1].

The path length between w1 and w2, can be determined from bellow of three cases:

1. Both the words w1 and w2 have same meaning so path length is zero.
2. If w1 and w2 are not in the same synset, but their synset contains one or more common words. For example, in Fig. the synset for boy and girl contain the common word which is child so path length becomes one.
3. w1 and w2 are not in the same synset and their synset doesn't contain any common words, then the path length becomes:

$$f_1(l) = e^{\alpha l} \quad (16)$$

$\alpha$ is a constant and f1 is within the range from 0 to1. The Depth function is

$$f_2(h) = \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} \quad (17)$$

here β > 0 and it is a smoothing factor. As $\beta \to \infty$, the depth of a word in the semantic nets is not considered.

So the word similarity measure is calculated as:

$$s(w_1,w_2) = e^{-\alpha l}.\frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} \quad (18)$$

2. Semantic Similarity Between Sentences

$T_1$ and $T_2$ are two sentences, a joint word set is formed containing all the distinct words from $T_1$ and $T_2$.

$$T = T_1 \cup T_2 \qquad = \{w_1\ q_2 \dots w_m\}$$

Then the lexical semantic vector is formed from the joint word set, denoted by $\check{s}$. Each entry of the semantic vector corresponds to a word in the joint word set. The value of an entry of the lexical semantic vector, $\check{s}_i (i = 1,2,\dots,m)$, is determined by the semantic similarity of the corresponding word to a word in the sentence. The value of an entry of semantic vector is :

$$s_i = \check{s}.I(w_i).I(\widetilde{w_i}) \quad (19)$$

$I(w_i)$ and $I(\widetilde{w_i})$ Are information content. *wi* is a word in the joint word set, $\widetilde{w_i}$ is its associated word in the sentence. So semantic similarity between two sentences is defined as the cosine coefficient between the two vectors:

$$S_s = \frac{s_1.s_2}{||s_i||.||s_2||} \quad (20)$$

3. Word Order Similarity between Sentences

Word Order Similarity is denoted by $S_r$ and derived as

$$S_r = 1 - \frac{||r_1 - r_2||}{||r_1 + r_2||} \quad (21)$$

$r_1$ and $r_2$ are word order vectors for T1 and T2, respectively.

$$r1 = \{a1 \dots aj \dots aj + k \dots am\} \text{ for } T1;$$

$$r2 = \{b1 \dots bj \dots bj + k \dots bm\} \text{ for } T2$$

$a_j$ and $a_{j+k}$ are the entries for the considered word pair in $T_1$, bj and $b_{j+k}$ are the corresponding entries for the word pair in T2, and k is the number of words from $w_j$ to $w_{j+k}$.

4. Overall Sentence Similarity

The equation of overall sentence similarity is:

$$S(T_1,T_2) = \delta S_s + (1-\delta)S_r \quad (22)$$

$$S(T_1,T_2) = \delta \frac{s_1.s_2}{||s_i||.||s_2||} + (1-\delta)\frac{||r_1 - r_2||}{||r_1 + r_2||}, \quad (23)$$

where $\delta \leq 1$ decides the relative contributions of semantic and word order information to the overall similarity computation. It is combination of semantic similarity and word order similarity.

### 2.4 Omiotis

It is the 3 step process:
1-semantic link is constructed between all word senses in the WordNet and it will pre-computes a relatedness score.2-then semantic relatedness between word pair is derived by taking into account the relatedness of their corresponding WordNet senses. 3-then final semantic relatedness score is derived for two text segments extending word to word relatedness. [5]

1. Construct Semantic Links between Words

Constructing semantic network from the word thesauri is the primary step. For example the bellow figure shows the construction of a semantic network for two words ti and tj. This expansion process is repeated recursively until the shortest path between senses S.i.2 and S.j.1 is found. If there is no path found between senses then the words and senses are not semantically related.

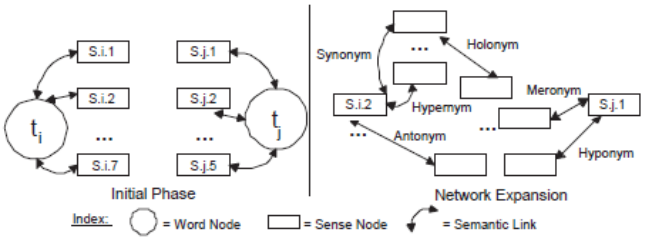


Fig 3. Constructing semantic networks from word thesauri.

Now semantic relatedness between pair of concepts:

**Def1** Word thesaurus O is given, a weight $w \in (0,1)$ is assigned by a weighting scheme for each edge, $S = (s_1, s_2)$ is a pair of senses, $l$ is a length of path $P$ connects two senses, then the semantic compactness of S *(SCM(S,O, P))* is defined by : $SCM(S,O,P) = \prod_{i=1}^{l} w_i$ where $w_1, w_2, \ldots, w_i$ are the weights of path's edges'. $SCM(S,O,P) = 1$ if $s_1 = s_2$. $SCM(S,O,P) = 0$ if there is no path between $s_1$ and $s_2$ then.

**Def2** Word thesaurus O given , a pair of senses $S = (s_1, s_2)$, where $s_1, s_2 \in O$ and $s_1 \neq s_2$, and a path $P =< p_1, p_2, \ldots, p_l >$ of length $l$, where either $s_1 = p_1$ and $s_2 = p_l$ or $s_1 = p_l$ and $s_2 = p_1$,the semantic path elaboration of the path *(SPE(S,O,P))*is defined as: $SPE(S,O,P) = \prod_{i=1}^{l} \frac{2d_i d_{i+1}}{d_i + d_{i+1}} \cdot \frac{1}{d_{max}}$, where $d_i$ is the depth of sense $p_i$ according to O, and $d_{max}$ the maximum depth of O. *If* $s_1 = s_2$, then $d_1 = d_2 = d$ and $SPE(S,O,P) = \frac{d}{d_{max}}$. If there is no path from $s_1$ *to* $s_2$ then $SPE(S,O,P) = 0$.

We combine these two measures following the definition of the *Semantic Relatedness* between two terms:

**Def3** Word thesaurus O is given, and *a* pair of senses $S = (s_1, s_2)$ the semantic relatedness of *S (SR(S,O))* is defined as $max_P \{SCM(S,O,P) \cdot SPE(S,O,P)\}$.

There is possibility that two senses can be connected by more than one semantic path. So for different path, the senses' compactness take different values. Here, for finding shortest path, Dijkstra's algorithm with some modification is used.

Next is to find semantic relatedness between pair of terms: Based on Definition 3, the semantic relatedness between a pair of terms $T(t_1, t_2)$ is calculated as follows.**Def4** Word thesaurus O given, let $T = (t_1, t_2)$ be a pair of terms for which there are entries in O, let $X_1$ be the set of senses of $t$ and $X_2$ be the set of senses of $t_2$ in O. Let $S_1, S_2, \ldots, S_{|X_1|.|X_2|}$ be the set of pairs of senses, $S_k = (s_i, s_j)$, with $s_i \in X_1$ and $s_j \in X_2$. Now the semantic relatedness of *T (SR(T, S,O))* is defined as:

$max_{Sk}\{max_P\{SCM(S_k,O,P) \cdot SPE(S_k,O,P)\}\} = max_{Sk}\{SR(S_k,O)\}$ k = 1..|X1| · |X2| for all k. Semantic relatedness between two terms $t_1, t_2$ where $t_1 \equiv t_2 \equiv t$ and $t \neq O$ is defined as 1. Semantic relatedness between $t_1, t_2$ when $t_1 \notin$ O and $t_2 \notin$ O, or vice versa, is considered 0.

2. Omiotis

Lexical relevance is denoted as $\lambda_{a,b}$ between terms $a \in A$ and $b \in B$ .To find lexical relevance based on the of the respective terms' harmonic mean, TF-IDF values, given by:

$$\lambda_{a,b} = \frac{2.TF\_IDF(a,A).TF\_IDF(b,B)}{TF_{IDF(a,A)} + TF\_IDF(b,B)} \quad (24)$$

$TF\_IDF(a,A)$ and $TF\_IDF(b,B)$ are used to measure the qualitative strength of a and b respective texts.

After that it will find for every word $a$ in text $A$ the corresponding word $b$ in text $B$ that maximizes the product of lexical similarity and semantic relatedness values

$$b_* = arg\max_{b \in B}(\lambda_{a,b}.SR(a,b)) \quad (25)$$

Here $b_*$ corresponds to that term in text $B$, entails the maximum semantic relatedness and lexical similarity with term $a$ from text $A$. Similarly, $a_*$,

$$a_* = arg\max_{a \in A}(\lambda_{a,b}.SR(a,b)) \quad (26)$$

Then aggregate the semantic and lexical relevance scores for all terms in text $A$, with reference to their best match in text $B$ which is denoted as shown in the Equation bellow:

$$\zeta(A,B) = \frac{1}{|A|}\left(\sum_{a \in A} \lambda_{a,b_*}.SR(a,b_*)\right) \quad (27)$$

Then find same thing from the words of B to the words of A, to cover the cases where the two texts do not have an equal number of terms. Finally, to find the degree of relevance between texts A and B combine the values estimated for

their terms that entail the maximum semantic and lexical relevance to one another, which is given by bellow equation:

$$\text{Omiotis}(A, B) = \frac{[\zeta(A, B) + \zeta(B, A)]}{2} \quad (28)$$

## 3. COMPARISON AND EXPERIMENTAL RESULTS

For comparing the performance of all the method , the dataset which commonly used for all measures and is produced by Li et al. (2006) is used. It contains 65 sentence pairs. For each sentence pair, similarity scores have been given by 32 human participants, ranging from 0.0 (the sentences are not related at all in meaning), to 1.0 (the sentences are completely related in meaning). From the 65 sentence pairs, the comparison among different measures is shown for 30 sentences pair. The pearson's and spearman's correlation is determined.

**Table -1:** Human, STS, SyMSS, STASIS and *Omiotis* scores for the 30 sentence pairs

| No. | Sentence Pair | Human | STS | SyMSS | STASIS | Omiotis |
|---|---|---|---|---|---|---|
| 1 | cord:smile | 0.01 | 0.06 | 0.32 | 0.33 | 0.17 |
| 5 | autograph:shore | 0.01 | 0.11 | 0.28 | 0.29 | 0.15 |
| 9 | asylum:fruit | 0.01 | 0.07 | 0.27 | 0.21 | 0.11 |
| 13 | boy:rooster | 0.11 | 0.16 | 0.27 | 0.53 | 0.31 |
| 17 | coast:forest | 0.13 | 0.26 | 0.42 | 0.36 | 0.30 |
| 21 | boy:sage | 0.04 | 0.16 | 0.37 | 0.51 | 0.24 |
| 25 | forest:graveyard | 0.07 | 0.33 | 0.53 | 0.55 | 0.30 |
| 29 | bird:woodland | 0.01 | 0.12 | 0.31 | 0.33 | 0.11 |
| 33 | hill:woodland | 0.15 | 0.29 | 0.43 | 0.59 | 0.50 |
| 37 | magician:oracle | 0.13 | 0.20 | 0.23 | 0.44 | 0.11 |
| 41 | oracle:sage | 0.28 | 0.09 | 0.38 | 0.43 | 0.11 |
| 47 | furnace:stove | 0.35 | 0.30 | 0.24 | 0.72 | 0.22 |
| 48 | magician:wizard | 0.36 | 0.34 | 0.42 | 0.65 | 0.53 |
| 49 | hill:mound | 0.29 | 0.15 | 0.39 | 0.74 | 0.57 |
| 50 | cord:string | 0.47 | 0.49 | 0.35 | 0.68 | 0.55 |
| 51 | glass:tumbler | 0.14 | 0.28 | 0.31 | 0.65 | 0.52 |
| 52 | grin:smile | 0.49 | 0.32 | 0.54 | 0.49 | 0.60 |
| 53 | serf:slave | 0.48 | 0.44 | 0.52 | 0.39 | 0.50 |
| 54 | journey:voyage | 0.36 | 0.41 | 0.33 | 0.52 | 0.43 |
| 55 | autograph:signature | 0.41 | 0.19 | 0.33 | 0.55 | 0.43 |
| 56 | coast:shore | 0.59 | 0.47 | 0.43 | 0.76 | 0.93 |
| 57 | forest:woodland | 0.63 | 0.26 | 0.50 | 0.70 | 0.61 |
| 58 | implement:tool | 0.59 | 0.51 | 0.64 | 0.75 | 0.74 |
| 59 | cock:rooster | 0.86 | 0.94 | 1.00 | 1.00 | 1.00 |
| 60 | boy:lad | 0.58 | 0.60 | 0.63 | 0.66 | 0.93 |
| 61 | cushion:pillow | 0.52 | 0.29 | 0.39 | 0.66 | 0.35 |
| 62 | cemetery:graveyard | 0.77 | 0.51 | 0.75 | 0.73 | 0.73 |
| 63 | automobile:car | 0.56 | 0.52 | 0.7 | 0.64 | 0.79 |
| 64 | midday:noon | 0.96 | 0.93 | 1.00 | 1.00 | 0.93 |
| 65 | gem: jewel | 0.65 | 0.65 | 0.36 | 0.83 | 0.82 |

**Table -2** Spearman's and Pearson's correlation

| | Spearman's ρ | Pearson's r |
|---|---|---|
| **STS** | 0.838 | 0.853 |
| **SyMSS** | 0.71 | 0.76 |
| **STASIS** | 0.8126 | 0.8162 |
| **Omiotis** | 0.8905 | 0.856 |

Above table1 shows the similarity scores of different methods with human similarity scores and table 2 represents Spaearman's and pearson's coefficients for STS, SyMSS, STASIS and Omiotis.

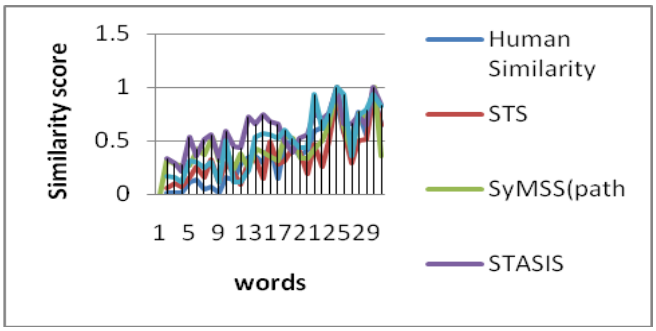


**Chart -1**: Graphical representation of similarity scores for different measures

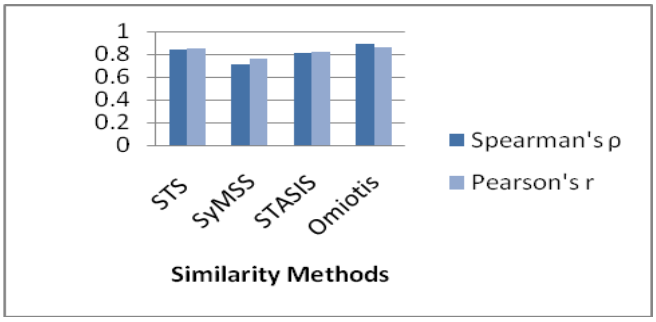


**Chart -2**: Comparison of Spearman and pearson's correlations for different measure

## 4. CONCLUSION

STS and SyMSS determines the similarity of two texts from the both semantic and syntactic information. The STS has the lower time complexity compared to others which is advantageous. In SyMSS the influence of adjectives and adverbs will help in the calculation of semantic similarity STASIS measures the semantic similarity between sentences or very short texts, based on semantic and word order information. Omiotis measure is very useful in the paraphrase recognition and sentence similarity. It will outperforms most of the above method in calculating semantic similarity.

## REFERENCES

[1] Jia Wei Chang , Ming Che Lee b, Tzone I Wang, "Integrating a semantic-based retrieval agent into case-based reasoning systems: A case study of an online bookstore," 2015 Elsevier,pp. 15–64.

[2] A. Islam, D. Inkpen, Semantic text similarity using corpus-based word similarity and string similarity, ACM Trans. Knowl. Discov. Data 2 (2) (2008) 1–25. [50] G. Tsatsaronis, I. Varlamis, M. Vazirgiannis, Text relatedness based on a word thesaurus, J. Artif. Intell. Res. 37 (1) (2010) 1–39..

[3] J. Oliva, J.I. Serrano, M.D. del Castillo, A´. Iglesias, SyMSS: a syntax-based measure for short-text semantic similarity, Data Knowl. Eng. 70 (4) (2011) 390–405.

[4] G. Tsatsaronis, I. Varlamis, M. Vazirgiannis, Text relatedness based on a word thesaurus, J. Artif. Intell. Res. 37 (1) (2010) 1–39.

[5] Y. Li, D. McLean, Z.A. Bandar, J.D. O'Shea, K. Crockett, Sentence similarity based on semantic nets and corpus statistics, IEEE Trans. Knowl. Data Eng. 18 (8) (2006) 1138–1150.

[6] R. Rada, H. Mili, E. Bicknell, M. Blettner, Development and application of a metric on semantic nets, IEEE Transactions on Systems, Man and Cybernetics 19 (1)(1987) 17–30.

[7] P. Resnik, Using Information Content to Evaluate Semantic Similarity in a Taxonomy, Proceedings of the 14th International Joint Conference on Artificial Intelligence, 1995, pp. 448–453.

[8] J. Jiang, D. Conrath, Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy, Proceedings on International Conference on Research in computational Linguistics, 1997, pp. 19–33.

[9] D. Lin, Using Syntactic Dependency as a Local Context to Resolve Word Sense Ambiguity, Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics, 1997, pp. 64–71.

[10] S. Banerjee, T. Pedersen, Extended Gloss Overlaps as a Measure of Semantic Relatedness, Proceedings of the Eighteenth International Joint conference on Artificial Intelligence, 2003, pp. 805–810.

S. Patwardhan, Incorporating dictionary and corpus information into a context vector measure of semantic relatedness, Master's thesis, University of Minnesota, Duluth, 2003.