MANAGING THE SYNTACTIC BLINDNESS OF LATENT SEMANTIC ANALYSIS

Raja Muhammad Suleman and Ioannis Korkontzelos

Department of Computer Science, Edge Hill University,
Ormskirk, Lancashire L39 4QP, United Kingdom

ABSTRACT

Natural Language Processing is a sub-field of Artificial Intelligence that is used for analysing and representing human language automatically. Natural Language Processing has been employed in many applications, such as information retrieval, information processing, automated answer grading etc. Several approaches have been developed for understanding the meaning of text, commonly known as semantic analysis. Latent Semantic Analysis is a widely used corpus-based approach that evaluates similarity of text on the basis of semantic relations among words. Latent Semantic Analysis has been used successfully in different language systems for calculating the semantic similarity of texts. However, Latent Semantic Analysis ignores the structural composition of sentences and therefore this technique suffers from the syntactic blindness problem. Latent Semantic Analysis fails to distinguish between sentences that contain semantically similar words but have completely opposite meaning. Latent Semantic Analysis is also blind to the syntactic structure of a sentence and therefore cannot differentiate between sentences and lists of keywords. In such a situation, the comparison between a sentence and a list of keywords without any syntactic structure gets a high similarity score. In this research we propose an algorithmic extension to Latent Semantic Analysis which focuses on syntactic composition of a sentence to overcome Latent Semantic Analysis's syntactic blindness problems. We tested our approach on sentence pairs containing similar words but having different meaning. Our results showed that our extension provides more realistic semantic similarity scores.

KEYWORDS

Natural Language Processing, Natural Language Understanding, Latent Semantic Analysis, Semantic Similarity.

1. INTRODUCTION

Natural Language Processing (NLP) is a sub-field of Artificial Intelligence concerned with the techniques of understanding and generating natural language by machines [1]. The goal of NLP is to achieve human-like language processing abilities. It is concerned with the interaction between computers and human languages. NLP tasks are broadly classified into two separate classes; Natural Language Understanding (NLU) and Natural Language Generation (NLG) [2]. NLU deals with the analysis of language for the purpose of producing a meaningful representation, while NLG deals with the construction of sentences from a given representation. NLU techniques have been used extensively in semantic search, information extraction, machine translation, text summarization and automated grading tasks.

Semantic similarity scores are used to compute the similarity between different texts. Many NLP tasks require a set of texts to be compared with another. Simple string-based scoring fails in such complex scenarios because a word might have many synonyms that may be used

Dhinaharan Nagamalai et al. (Eds): NATP, ACSTY, ADCOM, SOFE, CCCIOT, MLSC, SVC - 2020
pp. 01-08, 2020. © CS & IT-CSCP 2020 DOI: 10.5121/csit.2020.100401

interchangeably throughout the text. To handle such variations a more robust processing technique needs to be used. Latent Semantic Analysis (LSA) is one such technique that allows to compute the semantic overlap between different pieces of text. The underlying concept of LSA is that the meaning of text is related to existence and non-existence of distinct words [3]. LSA considers that the words with similar meaning will occur in a similar context. LSA has been used successfully in diverse NLP applications [4,5,6]. Even though LSA provides robust results in different applications, it is still prone to errors due to certain shortcomings in its basic architecture. LSA is affected by the following inherent problems:

1. LSA works on semantic relations between words and ignores the syntactic composition of sentences, which results in a high semantic similarity score between two sentences that might have a completely different meaning [7].
2. LSA does not consider the relationship between the subjects and objects during sentence comparison. For example, two sentences; "The boy stepped on a spider", and "The spider stepped on a boy" are considered equivalent. LSA gives a 100% semantic similarity score for these sentences. However, it can be seen that these two sentences are totally opposite of each other in their meaning.
3. LSA considers a list of words without having a proper sentence structure as a complete sentence [8]. For example, "boy spider stepped" is considered equivalent to above two sentences and LSA assigns a score of 100% whenever these sentences are compared.
4. LSA cannot differentiate between two sentences that are semantically related to each other, but one has a negation in it. For example; "Christopher Columbus discovered America" and "Christopher Columbus did not discover America". Negation completely changes the meaning of these sentences, however LSA gives more than 90% semantic similarity score for above two sentences.

2. EXTENDED LATENT SEMANTIC ANALYSIS (XLSA)

2.1. xLSA Overview

Latent Semantic Analysis has been successfully used as an Information Retrieval technique in both industrial and academic applications [15,16,17]. LSA uses a Bag-of-Words (BoW) model to generate a term-document matrix and then performs matrix decomposition using Singular Value Decomposition (SVD) which results in document and term vectors. Similarity measures such as cosine similarity are then applied to these vectors to evaluate the similarity of different documents. Since LSA relies on the existence and non-existence of terms in the documents and does not take into consideration the syntactic nature of these terms, semantic scores can be unpredictable in certain situations. To overcome this syntactic blindness of LSA, we propose an *algorithmic extension* which can be used with any LSA implementation to enhance the accuracy of the similarity results. Our approach adds syntactic information to the terms in the documents which is then used to complement the results of the LSA comparison. Each term is tagged with its corresponding Part-of-Speech tag and the Sentence Dependency Structure is used to highlight the role it plays in the context of the current sentence. This information is used to enhance the term matrix which is then used to compute the similarity scores. Figure 1 shows the algorithm for our approach.

2.2. xLSA Method

The proposed algorithm has been developed for English language and validated over a test set of sentences collected from different English language corpuses [9,10]. A sentence has one independent class that contains subject, verb and a complete thought. A complete thought must have a subject, a predicate and an optional object. Subject is "a person, thing or place that is

performing some action". Verb is the "description of an action" by the subject. Whereas Object of a sentence is; "a noun or pronoun that can be affected by the action of a subject". Sentences in English follow the Subject-Verb-Object (SVO) rule, where the verb shows the relationship between the subjects and objects in the sentences. xLSA uses Sentence Dependency Structure (SDS) and Part-of-Speech (POS) tags to identify the Subjects, Verbs and Objects in a sentence. This information is used to calculate the similarity between 2 sentences on the basis of matching the SVO structure. xLSA works in 2 phases; i) Pre-processing Phase and, ii) Evaluation Phase. The Pre-processing Phase decomposes the input sentences into atomic tokens and adds POS tags to each token. This phase also creates an SDS for the given sentences which is used in the Evaluation phase to determine structural similarity between the given sentences.

```
Step-1: Read Input Sentences
Step-2: Generate POS for sentences
Step-3: Generate SDS for sentences
Step-4: Decompose Sentences
Step-5: Repeat until end of tokens
          IF(A[t] = VERB_TAGS [] && A[w] != HELPING_VERB)
             VERB ← A[n]
          IF(A[t] = VERB_TAGS [] && A[w]=HELPING_VERB)
             Helping VERB ← A[n]
          IF(A[t] = 'IN' && A[w]='by')
             IN ← A[n]
          IF(A[t] = 'RB' && A[w]='not')
             SVO['N'] ←1
          IF(A[t] = 'DT' && 'IN' || A[t] = 'DT' && HELPING_VERB || A[t] = 'IN'
           && HELPING_VERB || A[t] = HELPING_VERB || count(S1) ==count(S2))
             SVO['P'] ←1
          ENDIF
        END Repeat
Step-6: IF (SVO[P]! = 1)
          Gaming flag ← 1
          Exit
        ENDIF
Step-7: Apply stemming on SVO and compare S1=S2 && V1=V2 && O1 =O2
Step-8: IF(Subject similarity < 0.4 && objects similarity<0.4)
              Cross compare S1 = O2 && O1 = S2
              IF(Cross compare > 0.7 && verb similarity>0.5)
                    Inverse flag ← 1.
        ENDIF
Step9:  Compute xLSA similarity
        Score = Avg(SubSubScore, VrbVrbScore, ObjObjScore)
        IF(Score > 0.7)
              IF( SVO['n'] = 1)
                    Return Negation Flag

Step-10: Return Score, Inverse flag, Negation flag
```

Fig. 1. xLSA Algorithm

After decomposition, both sentences are compared on the basis of subject, verb and object [11,12]. Before comparison, stemming operation is applied on subjects, verbs and objects. Stemming reverts words to their base form for easy comparison between different inflections of words [7]. For example, base form of words "Processing" and "Processed" is "Process". Stemming is applied to simplify the tracing of synonyms, i.e. both the words processing and processed are transformed to the base word process. This allows for quick search and

comparison without the need to check different forms of each word. After stemming xLSA performs a cross-comparison, i.e. it compares subject of first sentence with subject of second sentence, verb of first sentence with verb of second sentence and object of first sentence with object of second sentence. If first sentence has an object and the second sentence does not have an object than similarity score of objects is set to 0. For computing similarity of subject, verb and object it is necessary that they exist in both sentences, if they only exist in one sentence than their similarity score by default is set to 0. To compute the similarity we used the UMBC STS (Semantic Textual Similarity) Service API [18], which has been trained on the Stanford WebBase corpus. UMBC STS uses a hybrid approach by combining distributional similarity and LSA to compute word similarity. The UMBC service was evaluated on different words to determine the upper and lower bounds of acceptance thresholds. Similarity scores of < 0.4 (40%) were observed for words that were completely different and had no semantic relation, whereas similarity scores of > 0.7 (70%) were seen for words that were semantically or contextually similar.

If subject-to-subject similarity score of two sentences is less than minimum threshold (40%) and object-to-object similarity score of two sentences is also less than minimum threshold, then xLSA compares the subject of first sentence to object of second sentence and object of first sentence to subject of second sentence. If cross similarity score for both subject and object is greater than threshold (70%) and verb similarity score of two sentences is also greater than the threshold then xLSA sets inverse flag to '1' for the pair of sentences. Figure 2 shows the execution flow of the proposed scheme.

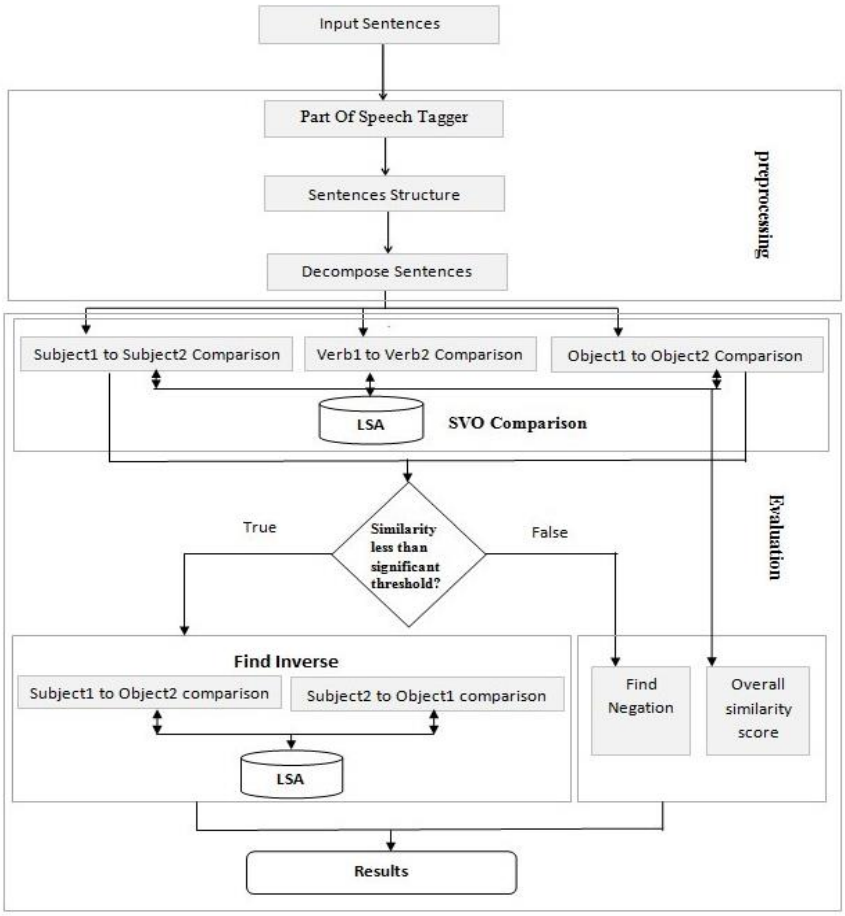

Fig. 2. xLSA Execution Flow

xLSA similarity score is calculated on the basis of subject, verb and object similarity score by using average method [13]. xLSA similarity score provides measures of semantic and syntactical similarity of both sentences. The scores are averaged with respect to the number of subjects, objects and verbs that exist in the sentences. Negative sentence states that concept is not true or incorrect. A negative adverb is used in order to deny the validity of a sentence. In order to claim that concept in the sentence is not true, a negation word 'not' is often added after first auxiliary verb in a positive sentence. xLSA finds negation in sentences that are semantically related. If the similarity score is greater than maximum threshold and inverse flag is 0, then xLSA checks negation in both sentences. POS tagger tags negative words with RB tag so xLSA finds RB tag in the sentence and checks corresponding value of that tag. If one of the sentences has negation then xLSA sets negation flag to 1. Negation flag is set to 0 when both sentences have negation or neither have negation in them.

3. RESULTS

Due to the subtle ambiguities in natural language, the results of LSA can be unpredictable as sentences that have a completely different meaning might be given a high similarity score under LSA. To evaluate our approach we used xLSA (with syntactic enhancements) and standard LSA (without syntactic enhancements) on the same set of sentences. The results showed that xLSA handles sentences which have similar words but different meaning more efficiently than standard LSA. Figure 3 shows the comparison between LSA and xLSA on sentences from the test set.

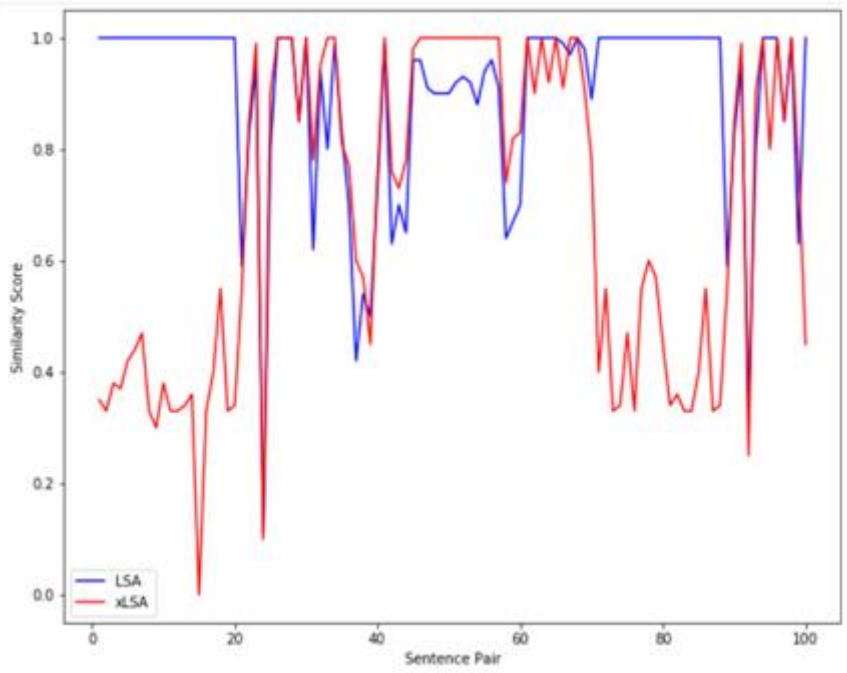

Figure 3. Standard LSA & xLSA scores on the test sentences

LSA gives a 100% semantic similarity score to all the sentences which have similar words irrespective of the effect they have on the meaning of a sentence. xLSA has been designed to calculate the semantic similarity between sentences not only on the basis of the similar words, but it also considers the syntactic structure of the sentences and the positioning of words in the sentences. This allows xLSA to distinguish between sentences that are semantically related on the surface level based on the words they contain but convey a completely different meaning. Table 1 shows some example sentences which contain same words but their meanings are quite

different. Standard LSA gives a 1 (100%) similarity score for all of these sentences whereas xLSA scores represent a much lower semantic similarity.

LSA doesn't consider the impact of negation on the meaning of a sentence, therefore it fails to identify when a sentence negates the other. Using xLSA, all the sentence pairs in the test dataset that contained at least one negation sentence were identified successfully. This means that two sentences might have a high semantic relatedness score, since they have common words, however, if one of the sentences, negates the other, then the semantic similarity between such sentences should be adjusted to highlight this fact.

Table 1. Inverse Sentences Similarity scores

Sentence Pair	LSA Score	xLSA Score
the earth must revolve around the sun. the sun must revolve around the earth.	1	0.55
koko was asked to choose a house or a tree. a house or a tree were asked to choose koko.	1	0.34
money cannot buy happiness. happiness cannot buy money.	1	0.36
the hard disk stores data. the data stores hard disk.	1	0.42
the cat climbs on a tree. the tree climbs on a cat	1	0.44
the dog bit a child. the child bit a dog.	1	0.47
tom is writing a letter and a book. letter and book are writing tom.	1	0.33

LSA doesn't consider the syntactic structure of the sentence during comparison. This means that a complete sentence when matched with a list of words can yield a similarity score as high as 100%. This might be counter-intuitive for applications that require proper sentences to be matched, e.g. automated answer grading systems. To overcome this, xLSA validates a proper syntactic structure to ensure that the input isn't only a list of keywords.

4. CONCLUSION

Latent Semantic Analysis (LSA) is corpus-based approach that computes similarity of text with a corpus using algebraic technique. LSA is used in document classification, semantic search engines, automated short answers grading, and many more tasks. LSA-based evaluation possesses strong correlation with human grading results [14]. LSA considers the semantic relationship among words while it overlooks the structure of a sentence, which might cause a logically wrong answer to be treated as a correct answer. Syntax plays a key role in understanding the meaning of a sentence and traditional LSA is blinded to this.

To mitigate LSA's syntactical blindness problem, this research aimed to provide an extension to simple LSA (xLSA) – which focuses on syntactic composition as well as semantic relations of sentences. xLSA analyses sentences for the identification of proper sentence structure using Sentence Dependency Structures (SDS) and the positioning of Part-Of-Speech (POS) tags. If the sentences have proper structure then xLSA focuses on dependency structures of sentences and

decomposes each sentence into Subject, Verb and Object (SVO). The sentences are compared on the basis of the similarity scores between the SVOs. xLSA is capable of identifying inverse sentences on the basis of cross comparison of subjects and objects of two sentences. xLSA also identifies negation in a pair of sentences that are semantically related to each other but where one of the sentence negates the other. We have trained and tested xLSA with semantically similar sentences from two corpuses [9,10]. Even though the results of using xLSA on the test sentences are promising, we do understand that our findings are limited by the number and categories of sentences that was used in testing the system. We aim to address these limitations in our future work by increasing the types of sentences our system can handle.

ACKNOWLEDGMENT

This research work is part of the TYPHON Project, which has received funding from the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No. 780251.

REFERENCES

[1] Khurana, Diksha & Koli, Aditya & Khatter, Kiran & Singh, Sukhdev. (2017). Natural Language Processing: State of The Art, Current Trends and Challenges.

[2] Liddy, Elizabeth D. "Natural language processing." (2001).

[3] Evangelopoulos, Nicholas, Xiaoni Zhang, and Victor R. Prybutok. "Latent semantic analysis: five methodological recommendations." European Journal of Information Systems 21.1 (2012): 70-86.

[4] Vrana, S.R., Vrana, D.T., Penner, L.A., Eggly, S., Slatcher, R.B. and Hagiwara, N., 2018. Latent Semantic Analysis: A new measure of patient-physician communication. Social Science & Medicine, 198, pp.22-26.

[5] Wegba, K., Lu, A., Li, Y. and Wang, W., 2018, March. Interactive Storytelling for Movie Recommendation through Latent Semantic Analysis. In 23rd International Conference on Intelligent User Interfaces (pp. 521-533). ACM.

[6] Jirasatjanukul, K., Nilsook, P. and Wannapiroon, P., 2019. Intelligent Human Resource Management Using Latent Semantic Analysis with the Internet of Things. International Journal of Computer Theory and Engineering, 11(2).

[7] Cutrone, Laurie, and Maiga Chang. "Auto-Assessor: Computerized assessment system for marking student's short-answers automatically." Technology for Education (T4E), 2011 IEEE International Conference on. IEEE, 2011.

[8] Braun, D., Hernandez-Mendez, A., Matthes, F. and Langen, M., 2017, August. Evaluating natural language understanding services for conversational question answering systems. In Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue (pp. 174-185).Lee, S.hyun. & Kim Mi Na, (2008) "This is my paper", ABC *Transactions on ECE*, Vol. 10, No. 5, pp120-122.

[9] Bowman, Samuel R., et al. "A large annotated corpus for learning natural language inference." arXiv preprint arXiv: 1508.05326 (2015).

[10] Young, Peter, et al. "From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions." Transactions of the Association for Computational Linguistics 2 (2014): 67-78.Gizem, Aksahya & Ayese, Ozcan (2009) Coomunications *& Networks*, Network Books, and ABC Publishers.

[11] Adhya, Soumajit, and S. K. Setua. "Automated Short Answer Grader Using Friendship Graphs." Computer Science and Information Technology-Proceedings of the Sixth International Conference on Advances in Computing and Information Technology (ACITY 2016). Vol. 6. No. 9. 2016.

[12] Ab Aziz, Mohd Juzaiddin, et al. "Automated marking system for short answer examination (AMS-SAE)." Industrial Electronics & Applications, 2009. ISIEA 2009. IEEE Symposium on. Vol. 1. IEEE, 2009.

[13] Wiemer-Hastings, Peter, and Iraide Zipitria. "Rules for syntax, vectors for semantics." Proceedings of the 23rd Annual Conference of the Cognitive Science Society. 2001.

[14] Gutierrez, Fernando, et al. "Hybrid ontology-based information extraction for automated text grading." Machine Learning and Applications (ICMLA), 2013 12th International Conference on. Vol. 1. IEEE, 2013.

[15] Nugroho, R., Paris, C., Nepal, S., Yang, J., & Zhao, W. (2020). A survey of recent methods on deriving topics from Twitter: algorithm to evaluation. Knowledge and Information Systems, 1-35.

[16] Chen, B. (2019). Latent Semantic Approaches for Information Retrieval and Language Modeling. Department of Computer Science & Information Engineering, National Taiwan Normal University, July. Accessed, 11-23.

[17] Shen, C. W., & Ho, J. T. (2020). Technology-enhanced learning in higher education: A bibliometric analysis with latent semantic approach. Computers in Human Behavior, 104, 106177.

[18] Lushan Han, Abhay L. Kashyap, Tim Finin, James Mayfield and Johnathan Weese, UMBC_EBIQUITY-CORE: Semantic Textual Similarity Systems, Proc. 2nd Joint Conf. on Lexical and Computational Semantics, Association for Computational Linguistics, June 2013.