Graph-Structures Matching for Review Relevance Identification

Lakshmi Ramachandran and **Edward F. Gehringer**
North Carolina State University
{lramach, efg}@ncsu.edu

Abstract

Review quality is determined by identifying the relevance of a review to a submission (the article or paper the review was written for). We identify relevance in terms of the semantic and syntactic similarities between two texts. We use a word order graph, whose vertices, edges and double edges help determine structure-based match across texts. We use WordNet to determine semantic relatedness. Ours is a lexico-semantic approach, which predicts relevance with an accuracy of 66% and f-measure of 0.67.

1 Introduction

Reviews play a critical role in making decisions, e.g., for grading students, accepting manuscripts for publication, or funding grants. Therefore, we must ensure that the decision-making party finds the review's content useful. Kuhne *et al.* (2010) found that authors were contented with reviewers who made an effort to understand their work. Nelson and Schunn (2009) found that reviews locating problems in the author's work, or providing suggestions for improvement help authors understand and use feedback effectively.

We investigated peer reviews from Expertiza, a web-based collaborative learning application (Gehringer, 2010). We found that reviewers provide comments such as, "Yes, it is good! It is very well organized." Such a review does not contain any unique information, or reference a specific concept or object in the author's submission. Such a generic review could work for *any* submission. Consider the comment, "I felt that some of the examples were clichéd." The reviewer criticizes the "examples" in the author's work but does not explain why they find the example "clichéd".

A review's quality may be assessed with the help of several metrics such as relevance of a review to the submission, its content type, coverage, tone, quantity of feedback provided (Ramachandran, 2011). In this paper we focus on the study of one review quality metric - *review relevance*.

A *relevant review* paraphrases the concepts described in a submission, with possible descriptions of problems identified in the author's work. Our aim is to identify whether a review is relevant to the work it was written for.

While paraphrasing, an idea may be restated by the reviewer with possible lexical and syntactic changes to the text. According to Liu *et al.* (2009), a good paraphrase, while preserving the original meaning of the text should contain some syntactic changes. According to Boonthum (2004) patterns followed commonly while paraphrasing include lexical synonymy, change in voice and change in sentence structure. Therefore, conventional text matching approaches, which look for exact matches, may not be good at identifying relevance.

2 Definition of Relevance

Definition Let S be the set of sentences in the text under review (the submission) and R be the set of review sentences. Let s and r represent a sentence in the submission and review respectively.

$$relevance(S,R) = \frac{1}{|R|}\sum_{\forall r \in R}\{\underset{\forall s \in S}{\text{argmax}}(\textit{ lexicoSemSim(s, r)})\} \quad (1)$$

lexicoSemSim(s,r) represents the *lexico-semantic* match between s and r. Relevance is the average of the best lexico-semantic matches of a review's sentences with corresponding submission sentences. The meaning and usage of *lexicoSemSim* has been explained in detail in Section 6. We acknowledge that all review sentences may not have corresponding matches in the submission. Our aim is only to identify the proportion of review text that is lexico-semantically relevant to a submission.

Since our aim is to identify the lexico-semantic match between texts, we need a representation that captures the syntax or order of tokens in a text. Hence we use a word order graph. Word order graphs are suited for identifying lexical and voice changes, which are common among paraphrased text. Similarity should capture the degree of relatedness between texts. Hence we use a WordNet-based metric (Fellbaum, 1998).

Figure 1 contains a sample submission and three sample reviews. The first review has some instances of exact match with the submission and is therefore relevant to the submission. However, the relevance of the second review may not be determined by a text overlaps

Proceedings of the TextGraphs-8 Workshop, pages 53–60,
Seattle, Washington, USA, 18 October 2013.

Submission: The debate on internet radio centers around an initiated fee imposed upon the internet radio stations. Proponents of the fee claim that it is necessary because artists are losing money since their music can be listened to without purchase. This fee, the internet radio stations contend, will drive them out of business. Though many artists see the internet stations as a welcome marketing tool to get their music heard. The radio stations began a campaign that involved a day of silence to raise awareness and get users to write to congress in their support.

Relevant review (text overlaps with the submission): Start with a general article on Internet radio rather than a listing of stations. The list of stations can follow that.

Relevant review (lexico-semantically related to the submission): The article should include an article prominently featuring the artists' interests. Arguments imply that the fee would bankrupt the stations.

Non-relevant: The article does seem to treat differing viewpoints fairly exploring both the negative consequences of hate speech and the negative consequences of censoring it.

Figure 1: The figure contains a sample submission, two relevant reviews – one with overt text matches and another that is lexico-semantically similar to the submission, and a non-relevant review.

match. The third review is lexico-semantically distinct from the submission.

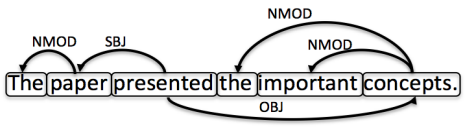
3 Related Work

There is little previous work in the area of identifying relevance between a review and a submission. Xiong and Litman (2011) use shallow metrics such as noun, verb count to identify review helpfulness. Their approach does not check for presence of paraphrases or summaries in a review. Ours is a pioneering effort in the application of relevance identification to the study of review helpfulness.

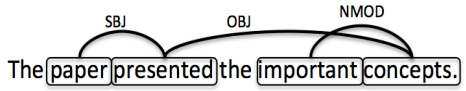
In this section we list some related work in the area of text matching, with a focus on approaches that use graphs such as lexical chains or dependency trees to represent text. Haghighi *et al.* (2005) use dependency trees to determine text entailment. They use node and path substitutions to compare text graphs.

Vertices in a dependency tree represent words, and edges capture the asymmetric dependency relationships between a head word and its modifier. Figure 2(a) contains a dependency tree representation (Bohnet, 2010) for the text "The paper presented the important concepts." We see that every token in the text is a vertex in the tree and edges depict governance relations (head → modifier). For example, "presented" is the root of this sentence and the edge between "presented" and "paper" signifies a subject relationship (SBJ). Dependency trees may not capture ordering information. For instance when we read the edges of the dependency tree in Figure 2(a) we get presented → paper, presented → concepts. The order of words in the edges is reversed, as in the case of presented → paper, although the actual order in the text is paper → presented.

The corresponding word order graph representation in Figure 2(b) captures the order of the words. The



(a) Dependency tree



(b) Word order graph

Figure 2: Displaying the ordering difference between a dependency tree representation and a word order representation for the text "The paper presented the important concepts."

word order graph captures SBJ—OBJ ordering as in paper—presented—concepts, which the directed edges in a dependency tree do not capture. Thus dependency tree representations may not be a useful representation in studying lexical or word order changes across documents.

Mani and Bloedorn (1997) suggest a graph search and matching approach for multi-document summarization. The graph matching approach used by Mani and Bloedorn focuses on concept or topics-based matching (noun entities). The graph captures adjacency relations between concepts or topics. Their graph representation does not capture ordering information, which would be suited for tasks involving comparison of lexical-order changes. As noted earlier, text matching with possible changes in word order is essential for a task like relevance identification. Existing representations and matching techniques do not capture this information. Van *et al.* (2009) construct phrase nets using regular expressions. Phrase nets are constructed for specific relations between tokens e.g. "X at Y" may indicate location of object X. Phrase nets are used as a tool for visualizing relations between objects in literary texts.

The document index graph (DIG) used by Hammouda and Kamel (2002), capture phrases of a document. Although the DIG captures order of words within a phrase, it does not capture the order of phrases within a document. As a result this representation does not capture complete sentence structure information, which may be necessary to identify whether a review contains sentence structure changes.

Mihalcea (2004) uses a graph to perform sentence extraction and summarization. Vertices in the graph represent sentences in a document. Weighted graph edges represent the degree of overlap across content of the sentences.

Kauchak and Barzilay (2006) suggest an automated technique to create paraphrases for human and

machine-translated text pairs, by substituting words in machine translated texts with their corresponding synonyms. They define paraphrases primarily in terms of synonyms of individual tokens.

Although there do exist independent research works that discuss graph-based summarization and paraphrasing techniques, they use content overlap or synonym matches to determine paraphrases. They do not consider context during text comparison. Our work is an amalgamation of existing research in the areas of text matching and paraphrase recognition.

4 Graph Representation

In a word order graph, edges represent relations between contiguous vertices. The graph captures word or phrase order of the text. Figure 2(b) contains the graph representation for a review.

A word order graph is suitable for applications that identify relevance or paraphrases across texts. Paraphrases may contain lexical changes and word or phrase shuffling across a text's length. Graph matches identify the presence or absence of lexical changes using the ordering and context that the word order graphs capture. A detailed description of the graph generation algorithm can be found in Ramachandran and Gehringer (2012).

1. The graph generator takes a piece of text as input and generates a graph as its output. We use period (.), semicolons (;) or exclamations (!) to break the text into multiple segments[1]. A text segment is a complete grammatical unit that can stand independent of the other clauses in the sentence in terms of its meaning.

2. The text is then tagged with parts-of-speech (POS) (NN, DT, VB, RB[2] etc.). We use the Stanford NLP POS tagger to generate the tagged text (Toutanova *et al.*, 2003). POS tags are useful in determining how to group words into phrases while still maintaining the order.

3. We use a heuristic phrase chunking technique[3] to group consecutive subject components (nouns, prepositions etc.) into a subject vertex, consecutive verbs (or modals) into a verb vertex, and similarly for adverb and adjective vertices. A graph vertex may contain a phrase or a token.

4. When a verb vertex is created the algorithm looks for the last created subject vertex to form an edge between the two. Ordering is maintained when an edge is created, i.e., if a subject vertex was formed before a verb vertex a subject—verb edge is created, else a verb—object edge is created. An adjective or an adverb is attached to the subject or verb vertex found in the sentence (i.e., subject—adjective or verb—adverb edge).

5. We tag graph edges with dependencies (Bohnet, 2010). We use the *anna* library available as part of the *mate tools* package to identify dependencies. Labels indicate the relation between words and their modifiers (e.g. SBJ – subject—verb relationship, OBJ – verb—object relationship). Post edge creation, we iterate through all edges to determine whether a dependency exists between the tokens representing the edge's vertices. We add an edge label if a dependency exists, e.g., "concepts—important" in Figure 2(b) captures the noun-modifier (NMOD) relation. Labels capture the grammatical role played by tokens in a text.

5 Semantic Relatedness

Match between two tokens could be one of: (1) exact, (2) synonym, (3) hypernym or hyponym (more generic or specific), (4) meronym or holonym (subpart or whole) (5) presence of common parents (excluding generic parents such as "object", "entity"), and (6) overlaps across definitions or examples of compared tokens[4], or (7) distinct or non-match. Each match is given a weight value, which represents its *degree of importance*, e.g., exact matches are more important than synonym matches, which are in turn more important than hypernyms or hyponyms and so on. Weight values are in the [0-6] range, 0 being the lowest match (distinct) and 6 the best match (exact). Unlike other approaches, which capture just exact or synonymy matches, our approach captures semantic relatedness between tokens using a few types of matches (Ramachandran, 2013).

Each match is identified using WordNet. WordNet has been used successfully to measure relatedness by Agirre *et al.* (2009). We use WordNet because it is faster than querying a knowledge source such as Wikipedia, which contains more than a million articles, not all of which may be relevant.

6 Lexico-Semantic Matching

The degree of match between two graphs depends on the degree of match between their vertices and edges. In this section we describe three types of matches across graphs - (1) phrase or token matching, (2) context matching, and (3) sentence structure matching. Figure 3 contains an overview of our relevance identification approach.

6.1 Phrase or token matching

In phrase or token matching, vertices containing phrases or tokens are compared across graphs. This

[1] Approach used is similar to that of the deterministic sentence splitter used by the Stanford NLP sentence splitter. `http://nlp.stanford.edu/software/tokenizer.shtml`

[2] NN - noun, DT - determiner, VB(Z) - verb, RB - adverb

[3] Our chunker groups words based on the POS tags without the overhead of training a model to perform chunking.

[4] Using context to match tokens was an approach used by Lesk (1986) for word-sense disambiguation.

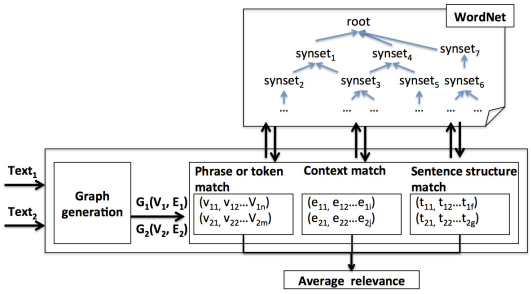


Figure 3: Overview of our approach for relevance identification task.

matching succeeds in capturing semantic relatedness between single or compound words. When vertices "concepts" and "points" are compared, a *common parents* match is found. This match would have been missed when using only an exact or synonym match.

$$\textit{Phrase(S, R)} = \frac{1}{|V_r|} \sum_{\forall r(v) \in V_r} \operatorname{argmax}_{\forall s(v) \in V_s} \{\textit{match}(s(v), r(v))\} \quad (2)$$

An overall phrase match is determined by taking the average of the best match that every review phrase has with a corresponding submission phrase. Similarity between two vertices is calculated as the average of matches between their constituent words or phrases. Match could be one of those listed in Section 5. In Equation 2, $r(v)$ and $s(v)$ refer to review and submission vertices respectively, and V_r and V_s is the set of vertices in a review and a submission.

6.2 Context matching

Context matching compares edges with same and different syntax, and edges of different types across two text graphs. We refer to the match as *context* matching since contiguous phrases (vertices) are chosen from a graph for comparison with another, i.e., more context. Relatedness between edges is the average of the vertex matches. We compare edge labels for matches retaining word order. Edge labels capture grammatical relations, and play an important role in matching. Hence if edges have the same labels then the average match is retained, else the match is halved. Some of the context-based matches include:

- **Ordered match** - Ordered match preserves the order of phrases in a text. We compare same type edges[5] with the same vertex order. Figure 4(a) shows the comparison of single edges from two review graphs. A match is identified between edges "important—concepts" and "necessary—points", because they capture the noun-modifier relationship (NMOD), and because a relation exists between tokens "concepts" and "points".

- **Lexical change** - Lexical match flips the order of comparison, e.g., we compare subject—verb with verb—object edges or vice versa. The match identifies paraphrases, which involve lexical changes. Figure 4(b) depicts lexical change match. When comparing edge "paper—presented" with edge "included—points", we compare vertex "paper" with "points" and "presented" with "included". A match is found between tokens "paper" and "points", resulting in the edge pair getting a relatedness value greater than a *non-match*. Had it not been for the lexical change match, such a relation may have been missed.

- **Nominalization match** - The match identifies noun nominalizations - nouns formed from verbs or adjectives (e.g. abstract → abstraction, ambiguous → ambiguity).

  In an ordered and lexical change match we compare same types of vertices (of the compared edges). We compare vertices of different types, e.g., the subject and verb vertices or the subject and adjective vertices. This match also captures relations between nouns and their adjective forms (e.g. ethics → ethical), and nouns and their verb forms (e.g. confusion → to confuse).

  In Figure 4(b) when we compare the edge "paper—presented" with edge "presentation—included", we compare "paper" (NN) with "included" (VB) and "presented" (VB) with "presentation" (NN) . Token "presentation" is the nominalization of token "presented", as a result of which a match is identified between the two edges.

$$\textit{Context(S, R)} = \frac{1}{3|E_r|}\Bigg(\sum_{r(e)\in E_r} \operatorname{argmax}_{\forall s(e)\in E_s}\{\textit{match}_{ord}(s(e), r(e))\} + \sum_{r(e)\in E_r} \operatorname{argmax}_{\forall s(e)\in E_s}\{\textit{match}_{lex}(s(e), r(e))\} + \sum_{r(e)\in E_r} \operatorname{argmax}_{\forall s(e)\in E_s}\{\textit{match}_{nom}(s(e), r(e))\}\Bigg) \quad (3)$$

In Equation 2, $r(e)$ and $s(e)$ refer to review and submission edges. The formula calculates the average best matches that review edges have with corresponding submission edges, for each of the above three types of matches $\textit{match}_{ord}$, $\textit{match}_{lex}$ and $\textit{match}_{nom}$. E_r and E_s represent the sets of review and submission edges respectively.

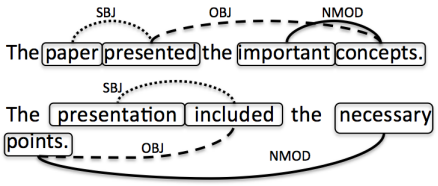
6.3 Sentence structure matching

Sentence structure matching compares double edges (two contiguous edges[6]), which constitute a complete segment[7] (e.g. subject—verb—object), across graphs.

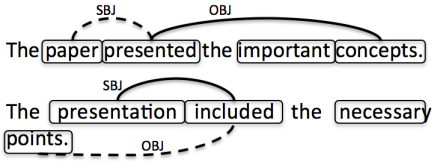
[5] Same type edges are edges with same types of vertices.

[6] Two consecutive edges sharing a common vertex.

[7] In this work we only consider single and double edges, and not more contiguous edges (triple edges etc.), for text matching.



(a) Ordered match - similar edges are compared across the two reviews, i.e., SBJ with SBJ, OBJ with OBJ etc.



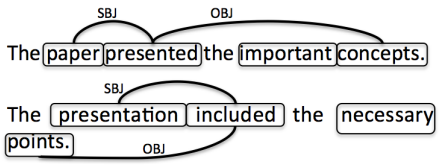
(b) Lexical change - edges of different types are compared, i.e., SBJ with OBJ and OBJ with SBJ respectively. Only the compared edges are shown in the graph representation.

Figure 4: Context matching across two text graphs.

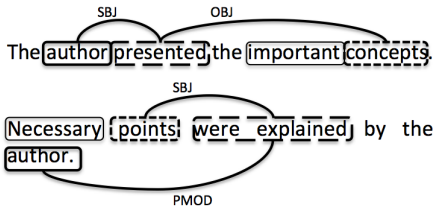
The matching captures similarity across segments, and it captures voice changes. Relatedness between double edges is the average of the vertex matches. Edge labels are compared in ordered matching, and the average vertex match is halved if the edge labels are different. Some sentence structure matches are:

- **Ordered match** - Double edges capture more word order than single edges, hence this matching captures more context. In Figure 5(a) double edges "paper—presented—concepts" and "presentation—included—points" are compared. Vertices "paper", "presented" and "concepts" are compared with vertices "presentation", "included" and "points" respectively.
- **Voice change** - Voice match captures word or phrase shuffling. Change of voice from active to passive, or vice versa is common with paraphrased text. Vertices of the same type are compared across double edges. However, the order of comparison is flipped. Consider the comparison between active and passive texts "The author presented the important concepts." and "Necessary points were explained by the author." in Figure 5(b). We compare "author" and "author" (exact match), "presented" and "were explained" (synonym match), and "concepts" and "points" (common parents match). This results in a cumulative voice match value of 4[8]. Only a voice change match succeeds in capturing such a relationship across the length of a sentence segment.

[8]Average of the vertex match values - 6 for exact match, 5 for synonym match, 2 for common parents match. Edge labels are not compared since the order of comparison of the vertices is flipped.



(a) Ordered sentence structure match.



(b) Voice change match - Order of comparison of the vertices is flipped, i.e., "author" is compared with "author", "presented" with "were explained" and "concepts" with "points".

Figure 5: Matching sentence segments across two text graphs. Compared vertices are denoted by similar borders.

$$\textit{SentStruct(S, R)} = \frac{1}{2|T_r|}\left(\sum_{r(t)\in T_r}\ \sum_{\forall s(t)\in T_s} \operatorname{argmax}\{\textit{match}_{ord}(s(t), r(t))\} + \sum_{r(t)\in T_r}\ \sum_{\forall s(t)\in T_s} \operatorname{argmax}\{\textit{match}_{voice}(s(t), r(t))\}\right) \quad (4)$$

The cumulative sentence structure match in Equation 3 calculates the average of the best ordered ($\textit{match}_{ord}$) and voice change ($\textit{match}_{voice}$) matches that a review's double edges have with corresponding submission double edges. $r(t)$ and $s(t)$ refer to double edges, and T_r and T_s are the number of double edges in the review and submission respectively.

Relevance in Equation 1 can be re-written as the average of the lexico-semantic relatedness values calculated from phrase, context and sentence structure matches.

$$\textit{relevance(S, R)} = \frac{1}{3}(\textit{Phrase(S, R)} + \textit{Context(S, R)} + \textit{SentStruct(S, R)}) \quad (5)$$

7 Experiments

We evaluate the performance of our graph matching approach in identifying the relevance of a review. We also study the performance of each match - *Phrase, Context* and *SentStruct* to determine whether the matches add value, and help improve the overall performance of our approach.

7.1 Data and method

We select review-submission pairs from assignments completed using Expertiza (Gehringer, 2010). Each review is compared with its respective submission, and

in order to include some explicit non-relevant cases reviews are compared with other submission texts. For the sake of evaluation we identify whether a review is relevant or not relevant to a submission. We chose 986 review-submission pairs containing an equal number of relevant and non-relevant reviews for our study. Relevance thresholds for the different matches are determined based on the averages. Two annotators labeled 19% of randomly selected data as *relevant* or *non-relevant*. We found an 80% agreement, and a Spearman correlation of 0.44 (significance $p < .0001$) between the two annotators' ratings. We use labels from the first annotator for testing due to the high percent agreement.

7.2 Results

Table 1 contains the accuracy and f-measure values of our approach in identifying relevance. A phrase or token matching contains no context. Consider the sample review "I would retitle 'Teaching, Using and Implementing Ethics' to 'Teaching and Using Codes of Ethics'." This review gets a good phrase match value of 3.3 with a submission (in Figure 1) discussing different codes of ethics. However, this review is not fully relevant to the content of the submission, since it is suggesting a change in title, and does not discuss the submission's content. Thus a simple non context-based phrase match tends to magnify the degree of relatedness between two texts. Thus although a phrase match is important, the lack of context may inflate relevance.

In the case of context matching, we found that lexical and nominalization matches produce lower match values than an ordered match. This happens because not all reviews contain word order changes or nominalizations, and flipping the order of matching results in a lower match when compared to that from an ordered match. The lower values decrease the average context matching, thus rendering a review non-relevant to a submission. This phenomenon explains the dip in context matching's accuracy and f-measure.

We observed a similar trend with sentence structure matches, where voice match produced a lower value than the ordered match in some of the cases. However the average *SentStruct* match in Equation 3, with an accuracy of 65%, shows an improvement over both phrase and context matches (Table 1).

Relevance is identified with an accuracy of 66% and f-measure of 0.67 (Table 1). Our approach has a high recall of 0.71, indicating a good degree of agreement with human relevance ratings. Thus the average of the phrase, context and sentence structure matches shows an improvement over each of the individual matches. This indicates that the addition of context (ordering) from edges and double edges contributes to an improvement in performance.

Dependency trees perform best for phrase matching (Table 1). Accuracy and f-measure of identifying relevance decreases for context, sentence structure and overall relevance matches. This is likely because edges in dependency trees capture only governance (head → modifier relation), and not word order.

Table 1: Comparing accuracy, precision, recall and f-measure values of our word order graph with those of a dependency-tree representation.

Metric	Phrase	Context	Sentence Structure	Relevance
Word order graph				
accuracy	64%	62%	65%	**66%**
precision	0.64	0.63	**0.65**	0.64
recall	0.67	0.60	0.63	**0.71**
f-measure	0.65	0.62	0.64	**0.67**
Dependency tree				
accuracy	64%	50%	52%	61%
precision	0.63	0.50	0.52	0.6
recall	0.7	0.40	0.41	0.65
f-measure	0.66	0.44	0.46	0.62

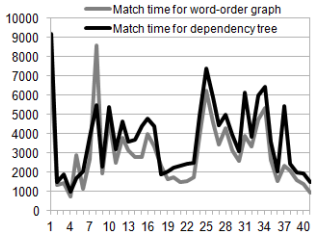


Figure 6: Identifying relevance with dependency trees takes more time (in milliseconds) than with word order graphs.

Dependency trees contain more vertices and edges than our graph, which results in an increase in the time taken to carry out pairwise comparison between the review and submission texts. We randomly selected 4% of the data to study the time taken to identify relevance by dependency trees, and by our graph. We found that in most cases dependency trees take more time than our graph (Figure 6). Thus our graph has a better performance, and is also faster than a dependency tree representation.

7.2.1 Comparison with a text overlap-based approach

We compare our approach with an overlap-based relevance identification approach. For this measure we consider the average of 1 to 4-gram overlaps between a review and a submission's texts to determine relevance. This is a precision-based metric, similar to the one used by Papineni *et al.* (2002).

$relevance_{overlap} = \frac{overlap(R,S)}{|R|}$, where *overlap* calculates the number of tokens in the review (R) that overlap with tokens in submission (S), and $|R|$ indicates the number of tokens in the review. Stopwords and frequent words are excluded from the numerator and denominator during overlap calculation.

This approach classifies a majority 62% of the records as non-relevant, and has an f-measure value of 0.59. The overlap approach has a high false negative

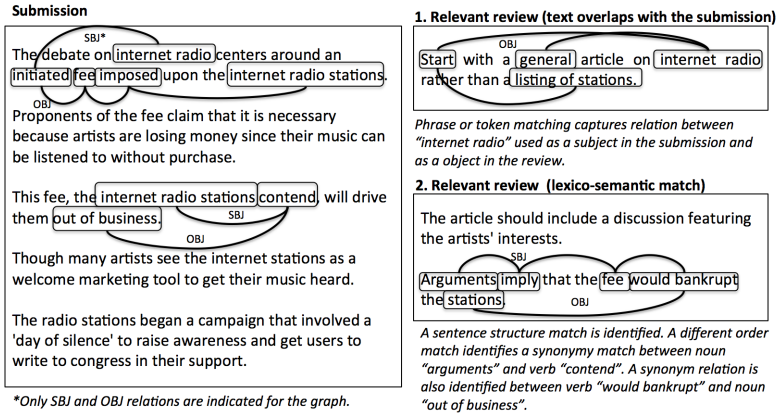


Figure 7: Example of phrase or token matching and sentence structure match between a review and a submission.

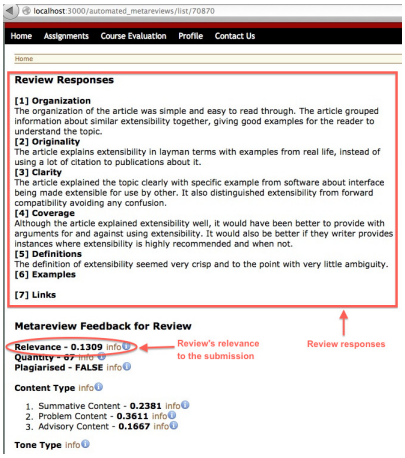


Figure 8: Output from our review assessment system displaying relevance value of reviews. Review's contents are relevant to article on "software extensibility".

rate i.e., several relevant reviews were wrongly classified as non-relevant (recall of 0.52). A simple text overlap, which does not capture the relations our approach succeeds in capturing, does not outperform our approach.

Figure 7 contains two sample reviews displaying phrase and sentence structure matching with sentences from a sample submission. The first review has some instances of exact match with the submission and its relevance may be easy to identify. However, relevance of the second review may not be determined by a text overlaps match. Our order-based matching and semantic relatedness metric help capture the relevance between the second review and the submission.

8 Feedback to Reviewers

A screenshot of the output from our review assessment system can be seen in Figure 8. In this example we have a review written for an article on *software extensibility*[9]. The sample review in Figure 8 has a relevance of 0.1309 (on a scale of 0–1). As can be seen from the screenshot, our automated assessment system provides feedback on not just relevance but on other metrics such as quantity, content and tone types too. However, a discussion of the approach involved in calculating each of these metrics is beyond the scope of this paper.

Our aim with this review assessment system is to motivate reviewers to update their review and make it more relevant to the text under review. This would help authors to better understand details of the review, and use the review to fix and improve their work.

In the future we are planning to improve the format of this output by providing textual feedback in addition to the numeric feedback. The feedback will point to specific instances of the review that need improvement. This may make it easy for reviewers to interpret the numeric score, and maybe further motivate reviewers to use the information to improve their reviews.

9 Conclusion

Assessment of reviews is an important problem in education, science and human resources, and so it is worthy of serious attention. In this paper we use a graph-based approach to determine whether a review is relevant to a piece of submission. Some important findings from our experiments are:

1. Additional context from graph edges and sentence structures helps improve the accuracy and f-measure of predicting relevance.

2. Our approach produces higher f-measure than a text overlap-based approach, that takes the average of 1 to 4-gram overlaps between review and submission texts to determine relevance.

[9]Software Extensibility `https://en.wikipedia.org/wiki/Extensibility`

3. Our approach produces higher accuracy and f-measure than dependency trees, which capture word-modifier information and not word order information.

References

Aria D. Haghighi, Andrew Y. Ng and Christopher D. Manning. 2005. Robust textual inference via graph matching. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing.* Vancouver, British Columbia, Canada 387–394.

Bernd Bohnet. 2010. Very high accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING).* Beijing, China. 89–97.

Bing Quan Liu and Shuai Xu and Bao Xun Wang. 2009. A combination of rule and supervised learning approach to recognize paraphrases. In *Proceedings of the International Conference on Machine Learning and Cybernetics.* July. 110–115.

Christiane Fellbaum. 1998. WordNet: An Electronic Lexical Database. *MIT Press, Cambridge, MA.*

Chutima Boonthum. 2004. iSTART: paraphrase recognition. In *Proceedings of the ACL 2004 workshop on Student research (ACLstudent).* Barcelona, Spain.

Conny Kuhne and Klemens Bohm and Jing Zhi Yue. 2010. Reviewing the reviewers: A study of author perception on peer reviews in computer science. *CollaborateCom.* 1–8.

David Kauchak and Regina Barzilay. 2006. Paraphrasing for automatic evaluation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL '06).* New York, New York. 455–462.

Edward F. Gehringer. 2010. Expertiza: Managing Feedback in Collaborative Learning. *Monitoring and Assessment in Online Collaborative Environments: Emergent Computational Technologies for E-Learning Support.* 75–96.

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and WordNet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Boulder, Colorado. 19–27.

Frank Van Ham, Martin Wattenberg and Fernanda B Viégas. 2009. Mapping text with phrase nets. *IEEE Transactions on Visualization and Computer Graphics* 15(6):1169–1176.

Inderjeet Mani and Eric Bloedorn. 1997. Multi-document summarization by graph search and matching. In *Proceedings of the fourteenth national conference on artificial intelligence and ninth conference on Innovative applications of artificial intelligence (AAAI '97).* Providence, Rhode Island. 622–628.

Khaled M. Hammouda and Mohamed S. Kamel. 2002. Phrase-based Document Similarity Based on an Index Graph Model. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM).*

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL).* Philadelphia, Pennsylvania. 311–318.

Kristina Toutanova, Dan Klein, Christopher D. Manning and Yoram Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of HLT-NAACL 2003*, 252–259.

Lakshmi Ramachandran and Edward F. Gehringer. 2011. Automated assessment of review quality using latent semantic analysis. *11th IEEE International Conference on Advanced Learning Technologies.* July. 136–138.

Lakshmi Ramachandran and Edward F. Gehringer. 2012. A Word-Order Based Graph Representation For Relevance Identification [poster]. *CIKM 2012, 21st ACM Conference on Information and Knowledge Management.* Maui, Hawaii. October.

Lakshmi Ramachandran and Edward F. Gehringer. 2013. An Ordered Relatedness Metric for Relevance Identification. In *proceedings of the Seventh IEEE International Conference on Semantic Computing (ICSC) 2013.*

Melissa M. Nelson and Christian D. Schunn. 2009. The nature of feedback: How different types of peer feedback affect writing performance. *Instructional Science.* 27(4):375–401.

Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on System documentation (SIGDOC).* Toronto, Ontario, Canada. 24–26.

Rada Mihalcea. 2004. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions (ACL demo)*. Stroudsburg, PA, USA.

Wenting Xiong and Diane Litman. 2011. Automatically predicting peer-review helpfulness. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers (HLT) - Volume 2.* Portland, Oregon. 502–507.