Sentence Similarity based on Dependency Tree Kernels for Multi-document Summarization

Şaziye Betül Özateş[*], Arzucan Özgür[*], Dragomir R. Radev[†]
[*]Department of Computer Engineering, Boğaziçi University
34342, Bebek, Istanbul, Turkey
{saziye.bilgin, arzucan.ozgur}@boun.edu.tr

[†]Department of EECS, University of Michigan
3917 Beyster Building, Ann Arbor, MI 48109
radev@umich.edu

Abstract

We introduce an approach based on using the dependency grammar representations of sentences to compute sentence similarity for extractive multi-document summarization. We adapt and investigate the effects of two untyped dependency tree kernels, which have originally been proposed for relation extraction, to the multi-document summarization problem. In addition, we propose a series of novel dependency grammar based kernels to better represent the syntactic and semantic similarities among the sentences. The proposed methods incorporate the type information of the dependency relations for sentence similarity calculation. To our knowledge, this is the first study that investigates using dependency tree based sentence similarity for multi-document summarization.

Keywords: multi-document summarization, sentence similarity, dependency grammars

1. Introduction

Multi-document summarization (MDS), which refers to the task of automatically generating a summary of multiple documents about the same topic without losing the most important information, is one of the most promising solutions proposed to overcome the information overload problem (Li et al., 2007).

Sentence similarity calculation is a crucial task for many extractive approaches to MDS (Erkan and Radev, 2004; Mihalcea and Tarau, 2005; Wang et al., 2008; Aliguliyev, 2009). Most of them use a bag of words model to compute sentence similarity. However, the bag of words model is sometimes inadequate for capturing the syntactic and semantic similarities among the sentences, which may affect the qualities of the summaries. To address this problem, we propose to employ dependency grammars, which represent the syntactic dependencies among the words in a sentence, for sentence similarity computation in MDS. Using this approach, concepts in multiple documents and relations between similar contents can be captured.

In this study, we first adapt two dependency tree based sentence similarity kernels in order to use them in MDS. These kernels are proposed by Culotta and Sorensen (2004) and Choi and Kim (2013) respectively, for relation extraction. They do not take the dependency relation types into account while calculating sentence similarity. We then propose a series of new sentence similarity kernels based on typed dependency grammars and test these kernels on LexRank (Erkan and Radev, 2004), a well-known and publicly available MDS system, by replacing its tf-idf based cosine similarity method with each of the kernels. The proposed similarity kernels make use of the binary dependency relations in the sentences. We conduct experiments on DUC 2003 and DUC 2004 Task 2 data sets. The results show that the best one of the proposed methods outperforms the other untyped tree kernels and LexRank's own sentence similarity method in terms of ROUGE-1 and ROUGE-2 scores.

2. Related Work

Several methods including supervised approaches (Das and Martins, 2007; Pei et al., 2012), topic driven models (Nastase, 2008; Hennig and Labor, 2009; Wang et al., 2009), and clustering based models (Radev et al., 2004; Aliguliyev, 2010) have been proposed in the literature for MDS. Recently, graph-based summarization methods have attracted the increasing attention of researchers (Erkan and Radev, 2004; Wan and Yang, 2008; Shen and Li, 2010) and have been successful when compared to the other state of the art summarization approaches (Mihalcea, 2004). Graph-based methods represent documents as a graph, where vertices are sentences and edges denote the similarity between the correponding pairs of sentences. LexRank (Erkan and Radev, 2004) is one of the most salient graph-based methods for MDS. Here the general idea is that sentences that have connections to many other significant sentences are considered to be important. Like most of the other graph-based studies, LexRank uses cosine similarity based on the tf-idf metric to measure the similarities among the nodes in a sentence graph. Yet, these methods treat sentences as bags of words. This representation may fail to capture some of the semantically related information, which in turn may affect the summary quality negatively.

We propose utilizing dependency grammars for sentence similarity computation in MDS. In the literature, dependency parsing has been used to find common information among sentences in order to perform sentence fusion (Barzilay and McKeown, 2005; Filippova and Strube, 2008) and to detect uninformative parts of sentences for the task of sentence compression (Yousfi-Monod et al., 2008; Blake et al., 2007). Dependency grammars have also been used for identifying concepts in specific domain terminologies by matching noun phrases to domain specific vocabularies (Fiszman et al., 2004), and for opinion summarization (Zhuang et al., 2006; Somprasertsri and Lalitrojwong, 2010). In addition, dependency parsing has been used to align sentences in documents with their human generated summaries in (Hirao et al., 2004) and to generate a dependency-based language model for Information Retrieval as in (Gao et al., 2004). To the best of our knowledge, none of the previous studies have used the dependency grammar concept to compute sentence similarity in a text summarization approach.

3. Methodology

Dependency tree representations of sentences allow us to utilize the syntactic dependency relations among words. Therefore, it is a more powerful approach than the bag of words representation for modeling the syntactic and semantic information in sentences. Considering this strength of dependency grammars, we first adapt two state-of-the-art dependency tree kernels (Culotta and Sorensen, 2004; Choi and Kim, 2013) originally proposed for relation extraction, to the MDS task. We refer to these kernels as the Dependency Tree Kernel (DTK) and the Dependency Trigram Kernel (Tri-K) throughout the paper. Next, we design a series of new sentence similarity methods based on typed dependency grammars. The following subsections describe the proposed dependency tree based similarity methods in detail.

3.1. Dependency Bigram Kernels

DTK and Tri-K do not take into account the types of dependencies in a sentence. They treat all dependencies in a dependency tree as having equal importance. However, there are different types of dependency relations in a tree and not all of them are equally important. For example, the dependency relation between a verb and its subject is more semantically significant than the dependency relation between a noun and its determiner for capturing the meaning of a sentence better.

We design a series of new methods that make use of typed dependency grammars to compute sentence similarity. Our methods use dependency tree *bigram units* and measure the similarity of two sentences using bigram unit matches. A bigram unit denotes a branch in the dependency tree, consisting of a *dependent* word, a *head* word, and the *type* of the dependency relation between them. For instance, the nodes *he* and *refused*, as well as the type of their dependency relation *nsubj* in the first sentence of Figure 1 form a bigram unit as $\{he, nsubj, refused\}$. The existance of similar bigram units in two sentences can give more clues about their semantic similarities.

We design four sentence similarity kernel methods which make use of these bigram structures.

Let us first define ${A_b}^i = \{{d_A}^i, {t_A}^i, {h_A}^i\}$ as the i^{th} bigram of sentence A where, ${d_A}^i$ is the dependent node, ${t_A}^i$ is the type, and ${h_A}^i$ is the head node of ${A_b}^i$. Then, the **Simple Approximate Bigram Kernel** (SABK) is defined as follows for the sentences A and B:

$$SABK(A, B) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} sim({A_b}^i, {B_b}^j)}{m+n} \quad (1)$$

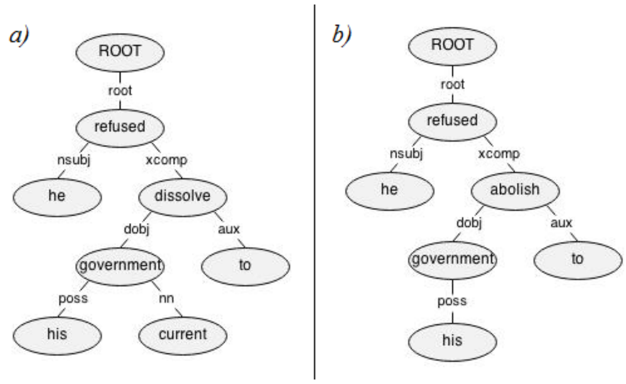

Figure 1: Typed dependency trees of the sentences *"He refused to dissolve his current government."* and *"He refused to abolish his government."*, respectively.

where m and n are the number of words in sentences A and B, respectively. The function *sim* is defined as:

$$sim({A_b}^i, {B_b}^j) = \left[(s({d_A}^i, {d_B}^j) + s({h_A}^i, {h_B}^j)\right] \times q({t_A}^i, {t_B}^j) \quad (2)$$

where *s* and *q* are binary functions:

$$s(a, b) = \begin{cases} 1, & \text{if } a = b \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

$$q(a, b) = \begin{cases} \theta, & \text{if } a = b \\ 1, & \text{otherwise} \end{cases} \quad (4)$$

and θ is a constant greater than 1 that determines the influence of a type match. The function *sim* gives partial scores to bigram unit comparisons, even if they do not totally match with each other. Bigram units that only have a common dependent word or a head word are considered as an approximate match. If their types also overlap, the similarity score is increased by a factor of θ.

SABK treats all words as having equal importance. However, this is hardly true for many cases. To model the importance of a word, we include the tf-idf (term frequency - inverse document frequency) values of the dependent and head words to the formula and form the **TF-IDF Based Approximate Bigram Kernel** (TABK) as defined below:

$$TABK(A, B) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} sim_t({A_b}^i, {B_b}^j)}{N(A) \times N(B)} \quad (5)$$

where $N(A)$ is the normalizer function:

$$N(A) = \sqrt{\sum_{i=1}^{n}(tf_{{d_A}^i}\ idf_{{d_A}^i})^2 + (tf_{{h_A}^i}\ idf_{{h_A}^i})^2} \quad (6)$$

and sim_t is defined as:

$$sim_t({A_b}^i, {B_b}^j) = \Big[(tf_{{d_A}^i}\ idf_{{d_A}^i} \times tf_{{d_B}^j}\ idf_{{d_B}^j}) \times s({d_A}^i, {d_B}^j) + (tf_{{h_A}^i}\ idf_{{h_A}^i} \times tf_{{h_B}^j}\ idf_{{h_B}^j}) \times s({h_A}^i, {h_B}^j)\Big] \times q({t_A}^i, {t_B}^j) \quad (7)$$

TABK does not encourage consecutive bigram matches that form a subtree in the dependency trees. However, a common subtree in the dependency trees of two sentences means that these sentences contain similar substructures. To emphasize this point, we design the **Matching Subtrees Kernel** (MSK) below:

$$MSK(A,B) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} sim_t({A_b}^i, {B_b}^j)}{N(A) \times N(B)} + \frac{\sum_{k=1}^{K}\sum_{l=1}^{L} s({d_A}^i, {d_B}^j) \times K_c(c_{{d_A}^i}(k), c_{{d_B}^j}(l))}{N(A) \times N(B)} \quad (8)$$

where c_x denotes the set of children nodes of the node x and $c_x(i)$ is the i^{th} child of the node x. Children kernel K_c is defined as:

$$K_c(n_i, n_j) = \begin{cases} \alpha s(n_i, n_j) + \nu K_c(a_i, b_j) \\ \forall a_i \in c_{n_i} \text{ and } \forall b_j \in c_{n_j}, \\ \text{if } d_i = d_j \text{ and } t_i = t_j \\ 0, \qquad \text{otherwise} \end{cases} \quad (9)$$

Here, ν is a decay factor to prevent high increase in the similarity score. In addition to comparing each bigram unit of the first sentence with each bigram unit of the second sentence, this kernel also tries to find matching subtrees by comparing the children nodes of matching bigrams. K_c recursively compares the children of a matching dependent word pair and gives a fixed score of α to matching children nodes of a matching bigram pair.

We also derive a **Composite Kernel** (CK) by combining tf-idf based kernels as follows:

$$CK(A,B) = \beta.TABK(A,B) + \delta.MSK(A,B) \quad (10)$$

where β and δ determine the influence of the corresponding kernel in the calculation of sentence similarity.

Type Abbreviation	Type Name
det	determiner
expl	expletive
goeswith	goes with
possessive	possessive modifier
preconj	preconjunct
predet	predeterminer
prep	prepositional modifier
punct	punctuation
ref	referent

Table 1: Unimportant dependency relation types.

4. Experiments and Results

4.1. Data Sets

We evaluated our methods on the Task 2 data sets of DUC 2003[1] and DUC 2004[2]. For the evaluation of the systems, the ROUGE[3] metric is used with the stemming option.

4.2. Experiments

Erkan and Radev (2004), evaluated their LexRank method using the MEAD summarization system, where they combined the LexRank method with Position and Length features, and used the Cross-Sentence Informational Subsumption (CSIS) reranker (Radev et al., 2004). Since we tested our similarity kernels in the LexRank system by replacing its own sentence similarity method, we used their environment in our experiments. We used the Dragon Toolkit[4], which is a development package for Information Retrieval and Text Mining (Zhou et al., 2007), to develop the dependency grammar kernels.

At the preprocessing phase, we first generated the dependency parse trees of the sentences in our data sets by using the Stanford Parser (Klein and Manning, 2003). Then, the feature set given in (Culotta and Sorensen, 2004) was created for each node in the tree. The *word, POS tag,* and *general POS tag* features were generated using the Stanford Parser. The *entity type* feature was created using the Stanford Named Entity Recognizer (NER) tool which is available in the Stanford CoreNLP Package[5].The *WordNet hypernyms* feature was generated using JAWS (Java API for WordNet Searching) (Spell, 2009). All words were stemmed using the Porter Stemmer (Porter, 1980).

For the bigram kernels, we filtered the dependency relation types in Table 1 as they did not improve the performances of the kernels in the experiments.

After these pre-processing steps, we ran the LexRank MDS method by setting the weights of both the continuous LexRank feature and the Position feature to 1. To stick with the experimental setup in (Erkan and Radev, 2004), we used the sentence length cutoff value of 9 and the CSIS reranker with the threshold value of 0.5. The results were

[1] http://duc.nist.gov/duc2003/tasks.html
[2] http://duc.nist.gov/duc2004/tasks.html
[3] http://www.berouge.com
[4] http://dragon.ischool.drexel.edu/
[5] http://nlp.stanford.edu/software/corenlp.shtml

All Systems	DUC 2003 Task 2		DUC 2004 Task 2	
	Rouge-1	Rouge-2	Rouge-1	Rouge-2
Simple Approximate Bigram Kernel (SABK)	0.3740	0.0954	0.3839	0.0946
TF-IDF based Apprx. Bigram Kernel (TABK)	0.3733	0.0964	0.3888	0.0957
Matching Subtrees Kernel (MSK)	0.3741	0.0985	0.3892	0.0955
Composite Kernel (CK)	0.3726	0.0970	0.3895	0.0964
Dependency Tree Kernel (DTK)	0.3611	0.0874	0.3689	0.0872
Dependency Trigram Kernel (Tri-K)	0.3611	0.0866	0.3721	0.0880
LexRank (tf-idf)	0.3673	0.0900	0.3832	0.0934
Lead-based	0.3590	0.0872	0.3666	0.0842
Random	0.3038	0.0473	0.3090	0.0447
Submodular Functions Approach	-	-	0.3890	-
Best sytem of DUC 2004	-	-	0.3822	-

Table 2: ROUGE-1 and ROUGE-2 scores of the bigram kernels, untyped dependency tree kernels, and baseline models in the LexRank system on DUC 2003 and DUC 2004 Task 2 data sets.

then compared with the original LexRank system that uses tf-idf based cosine similarity function as well as the Lead-based summarization approach, which selects sentences by using only the Position feature (Erkan and Radev, 2004), and the Random approach which composes a summary by selecting sentences in a random manner as the baseline approaches.

4.3. Results

Table 2 shows the ROUGE-1 and ROUGE-2 scores[6] for the experiments made for the bigram kernels, the best models of the untyped dependency tree kernels, and the baseline models on DUC 2003 and DUC 2004 Task 2 data sets. MSK outperforms all of the other methods on DUC 2003 although SABK shows almost the same performance with MSK according to ROUGE-1 scores. When we look at the ROUGE-2 scores of the methods, we observe that MSK reaches the best performance on DUC 2003. All of our approximate bigram kernels achieve better ROUGE-1 and ROUGE-2 scores than the untyped dependency tree kernels and LexRank's original similarity method. The best performance on DUC 2004 is reached by the composite kernel CK according to both ROUGE-1 and ROUGE-2 scores.

Our experimental results illustrate that detecting common subtrees in the dependency trees of two sentences leads to an increase in performance in terms of ROUGE-1 and ROUGE-2. Common subtree detection is used by our MSK and CK kernels and they both achieve the best results on the two data sets. It is also observed that including tf-idf values into the similarity calculation steps improves the results. This is due to the fact that the tf-idf measure is effective at highlighting the importance of a word.

The experimental results show that representing sentences as a set of their dependency bigram relations is a more effective approach than the bag-of-words representation model for sentence similarity computation in the MDS task. We compared the ROUGE-1 scores of our methods with the best system on DUC 2004 (Conroy et al., 2004) and one of the recent state-of-the-art methods, the submodular functions approach (Lin and Bilmes, 2011). Our MSK and CK kernels achieve similar performances with the submodular functions method and perform better than the best system on DUC 2004.

DTK and Tri-K obtain better performances than the lead-based method on both data sets. However, these untyped dependency tree based approaches failed to achieve higher scores than the original similarity method of the LexRank system.

5. Conclusion

We presented sentence similarity computation methods for MDS based on the dependency parse trees of the sentences. We adapted two different dependency tree based sentence similarity kernels, which have originally been proposed for relation extraction. We also proposed a number of new methods that make use of the typed dependency grammar representations of sentences. We evaluated these methods within the LexRank system and compared their performances with the original bag of words based sentence similarity method of LexRank. We showed that, although the untyped dependency tree based kernels DTK and Tri-K outperformed bag of words based kernels for relation extraction, they failed to achieve higher ROUGE-1 and ROUGE-2 scores than the bag of words based cosine similarity kernel in the task of MDS. All of the proposed sentence similarity methods outperformed the two untyped dependency tree kernels, LexRank's original similarity method, and the best system of DUC 2004. Similar performance with the state-of-the-art submodular functions approach is achieved. However, the improvement in the performance of LexRank by our typed dependency tree kernels is not found to be statistically significant. This might be due to the n-gram matching based nature of ROUGE. The limitations of the LexRank system might have also constrained these kernels from showing up their actual efficacy.

The proposed kernels can be integrated with other summarization frameworks that use sentence similarity computation. They can also be applied to other NLP tasks including relation extraction and question answering. We believe these kernels can lead to improvements in such systems, and will investigate this as future work.

[6]ROUGE version 1.5.5 with following options: -n 2 -m -w 1.2 -b 665 -c 95 -r 1000 -f A -p 0.5 -t 0 -2 4 -u -a -d

Our results demonstrate that the types of dependency relations are crucial for identifying the important parts of the sentences, and utilizing the dependency tree structures of sentences helps us to find similar substructures in them.

6. Acknowledgements

We gratefully acknowledge the support of this study by TÜBİTAK (The Scientific and Technological Research Council of Turkey) in the form of a graduate scholarship.

7. Bibliographical References

Aliguliyev, R. M. (2009). A new sentence similarity measure and sentence based extractive technique for automatic text summarization. *Expert Systems with Applications*, 36(4):7764–7772.

Aliguliyev, R. M. (2010). Clustering techniques and discrete particle swarm optimization algorithm for multi-document summarization. *Computational Intelligence*, 26(4):420–448.

Barzilay, R. and McKeown, K. R. (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Blake, C., Kampov, J., Orphanides, A. K., West, D., and Lown, C. (2007). UNC-CH at DUC 2007: Query expansion, lexical simplification and sentence selection strategies for multi-document summarization. In *Proceedings of the Document Understanding Conference 2007*.

Choi, M. and Kim, H. (2013). Social relation extraction from texts using a support-vector-machine-based dependency trigram kernel. *Information Processing & Management*, 49(1):303–311.

Conroy, J. M., Schlesinger, J. D., Goldstein, J., and O'leary, D. P. (2004). Left-brain/right-brain multi-document summarization. In *Proceedings of the Document Understanding Conference (DUC 2004)*.

Culotta, A. and Sorensen, J. (2004). Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423. Association for Computational Linguistics.

Das, D. and Martins, A. F. (2007). A survey on automatic text summarization. *Literature Survey for the Language and Statistics II course at CMU*, 4:192–195.

Erkan, G. and Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.(JAIR)*, 22(1):457–479.

Filippova, K. and Strube, M. (2008). Sentence fusion via dependency graph compression. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 177–185. Association for Computational Linguistics.

Fiszman, M., Rindflesch, T. C., and Kilicoglu, H. (2004). Abstraction summarization for managing the biomedical research literature. In *Proceedings of the HLT-NAACL Workshop on Computational Lexical Semantics*, pages 76–83. Association for Computational Linguistics.

Gao, J., Nie, J.-Y., Wu, G., and Cao, G. (2004). Dependence language model for information retrieval. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 170–177. ACM.

Hennig, L. and Labor, D. (2009). Topic-based multi-document summarization with probabilistic latent semantic analysis. In *Recent Advances in Natural Language Processing (RANLP)*.

Hirao, T., Suzuki, J., Isozaki, H., and Maeda, E. (2004). Dependency-based sentence alignment for multiple document summarization. In *Proceedings of the 20th international conference on Computational Linguistics*, page 446. Association for Computational Linguistics.

Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.

Li, S., Ouyang, Y., Wang, W., and Sun, B. (2007). Multi-document summarization using support vector regression. In *Proceedings of DUC*. Citeseer.

Lin, H. and Bilmes, J. (2011). A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 510–520. Association for Computational Linguistics.

Mihalcea, R. and Tarau, P. (2005). A language independent algorithm for single and multiple document summarization. In *Proceedings of IJCNLP*, volume 5.

Mihalcea, R. (2004). Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 20. Association for Computational Linguistics.

Nastase, V. (2008). Topic-driven multi-document summarization with encyclopedic knowledge and spreading activation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 763–772. Association for Computational Linguistics.

Pei, Y., Yin, W., Fan, Q., and Huang, L. (2012). A supervised aggregation framework for multi-document summarization. In Martin Kay et al., editors, *COLING*, pages 2225–2242. Indian Institute of Technology Bombay.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program: electronic library and information systems*, 14(3):130–137.

Radev, D. R., Jing, H., Styś, M., and Tam, D. (2004). Centroid-based summarization of multiple documents. *Information Processing & Management*, 40(6):919–938.

Shen, C. and Li, T. (2010). Multi-document summarization via the minimum dominating set. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 984–992. Association for Computational Linguistics.

Somprasertsri, G. and Lalitrojwong, P. (2010). Mining feature-opinion in online customer reviews for opinion summarization. *J. UCS*, 16(6):938–955.

Spell, B. (2009). Java api for wordnet searching (jaws).

Wan, X. and Yang, J. (2008). Multi-document summarization using cluster-based link analysis. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306. ACM.

Wang, D., Li, T., Zhu, S., and Ding, C. (2008). Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 307–314. ACM.

Wang, D., Zhu, S., Li, T., and Gong, Y. (2009). Multi-document summarization using sentence-based topic models. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 297–300. Association for Computational Linguistics.

Yousfi-Monod, M., Prince, V. V., et al. (2008). Sentence compression as a step in summarization or an alternative path in text shortening. In *Coling'08: International Conference on Computational Linguistics*, pages 137–140.

Zhou, X., Zhang, X., and Hu, X. (2007). Dragon Toolkit: Incorporating auto-learned semantic knowledge into large-scale text retrieval and mining. In *Proceedings of the 19 th IEEE International Conference on Tools with Artificial Intelligence (ICTAI*.

Zhuang, L., Jing, F., and Zhu, X.-Y. (2006). Movie review mining and summarization. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 43–50. ACM.