PyrEval: An Automated Method for Summary Content Analysis

Yanjun Gao, Andrew Warner and Rebecca J. Passonneau
The Pennsylvania State University
Pennsylvania, USA
yug125@cse.psu.edu, acw5456@psu.edu, rjp49@cse.psu.edu

Abstract

The pyramid method is a content analysis approach in automatic summarization evaluation for manual construction of a content model from reference summaries, and manual scoring of unseen summaries with the pyramid model. PyrEval automates the manual pyramid method. PyrEval uses low-dimension distributional semantics to represent phrase meanings, and a new algorithm, EDUA (Emergent Discovery of Units of Attraction), to solve a set cover problem to construct the content model from vectorized phrases. Because the vectors are pretrained, and EDUA is an efficient greedy algorithm, PyrEval can apply pyramid content evaluation with no retraining, and in excellent time. Moreover, PyrEval has been tested on many datasets derived from humans and machine generated summaries, and shown good performance on both.

Keywords: Content Evaluation, Summarization, Summarization Evaluation Tool

1. Introduction

Automatic summarization methods that produce a paragraph to express the main ideas of one or more texts have shifted in recent years. The long-standing prevalence of extractive summarizers, which select complete sentences from source documents, has begun to give way to abstractive summarizers, which rewrite or generate sentences to eliminate less important content (Bing et al., 2015; Rush et al., 2015; Liu et al., 2015; Durrett et al., 2016; Nema et al., 2017). Automatic content evaluation of summarization systems has not seen the same progress. ROUGE has long been the dominant method for evaluating summary content, and is used in most of the papers cited above. Here we present PyrEval, which applies pyramid content analysis (Nenkova and Passonneau, 2004), a method intended for evaluation of abstractive summarization. The main innovation in PyrEval is a novel formulation of the pyramid construction problem, and a greedy algorithm whose approximate solution is both efficient and effective.

The pyramid method, introduced in (Nenkova and Passonneau, 2004), groups the distinct ideas (content units) mentioned in reference summaries to create a pyramid model of content, and differentiates the content units by importance as well as meaning. It then evaluates target summaries against the pyramid content model. Early work applied manual annotation of the pyramid method for largely extractive automated summarizers at a series of Document Understanding Conferences (DUC) and Text Analysis Conferences (TAC) administered by the National Institute of Standards (NIST) from 2006 through 2011 (Passonneau et al., 2006; Varma et al., 2009; Ji et al., 2010). Figure 1 shows a content unit from one of the pyramids created for the NIST pyramid evaluation at DUC 2006, along with a sentence in a summary from a peer (automated summarizer) that was matched to the content unit (CU). The illustrated content unit, labeled by the annotator, lists a phrase from each of three reference summaries (D, E, H) that express the labeled meaning. The importance weight of the CU is the number of contributing reference summaries.

Automated pyramid methods such as PyrScore (Passonneau et al., 2013) or PEAK (Yang et al., 2016) have been used in recent work on abstractive summarization, as in (Bing et al., 2015) (PyrScore) and (Peyrard and Eckle-Kohler, 2017) (PEAK), but have disadvantages that PyrEval addresses. PyrScore requires manual construction of a pyramid content model, and PEAK depends on external resources that are computationally inefficient for large-scale use. This paper introduces the PyrEval software and gives an overview of how PyrEval builds on the earlier PyrScore. [1] Details of each component will be presented in other publications. Tests of PyrEval performance on five years of manual pyramid evaluations conducted by NIST show that average performance on all topics for a given year correlates well with manual pyramid.

Example Content Unit

Summary	**Label**: Black boxes record vehicle data (**Wt** = 3)
D1	"black boxes" that record data in the last seconds before a crash
E3	record data on vehicle performance ("black box")
H4	black boxes . . . record data pertinent to an auto crash to lead to safety improvements

Matched phrase from a peer summary

Peer	a device similar to an airliner's black box, . . . that automatically records the vehicle's

Figure 1: A content unit (CU) of weight 3 from a NIST manual pyramid for topic D0608, and a phrase from an extractive summary (peer) that was matched to the CU.

2. Related Work

The most widely used content evaluation tool for machine-generated summaries is ROUGE (Lin, 2004), which takes model summaries as references and scores target summaries by matching substrings. It correlates well with human scores in ranking performance of systems on multiple summarization tasks. But it is not reliable for evaluating a

[1] Availablefordownloadfromhttps://github.com/serenayj/PyrEval

single summary (Louis and Nenkova, 2009). In addition, it does not provide informative feedback on the ideas contained in a summary, or the important ideas that have not been mentioned. In contrast, pyramid content scores are reliable for individual summaries, the scores can be interpreted as indicating whether a summary contains mainly important ideas, and whether it contains enough of the important ideas. Finally, the scores can be justified with respect to specific ideas that are included or missing.

Interest in the pyramid method has been revived by development of PEAK (Yang et al., 2016), a fully automated version that performs well on student summaries. (Peyrard and Eckle-Kohler, 2017) incorporate PEAK scores into an optimization objective for generating summaries.

PEAK is one of two recent automated systems that implement pyramid evaluation (Yang et al., 2016). It constructs a pyramid from reference summaries, and scores target summaries against its automated pyramid. It extracts subject-predicate-object triples from reference summary sentences using ClauseIE (Del Corro and Gemulla, 2013), then assembles a hypergraph where each triple is a hyperedge containing three nodes: $subject$, $predicate$ and $object$. The semantic similarity of all pairs of nodes in distinct hyperedges is measured using Align, Disambiguate and Walk (ADW) (Pilehvar et al., 2013), which relies on WordNet (Miller, 1995). Triples from distinct summaries with high similarity are combined into content units. To assess a new summary, its triples are extracted, and the Munkres-Kuhn algorithm (Kuhn, 1955) is applied to a bipartite graph from content unit (CU) nodes u to nodes v that represent target summary triples.

PyrScore (Passonneau et al., 2013) uses manual pyramids to score target summaries, with Weighted Textual Matrix Factorization (WTMF) for latent semantic representation (Guo and Diab, 2012), and ngram segmentation to find all possible combinations of covering segmentations of input sentences. The scoring procedure can be formulated as a graph covering problem, where each sentence is a subgraph, and each segmentation assigned to a sentence is a vertex of the graph. The goal is to align a candidate sentence segmentation with content units from a manual pyramid via semantic similarity between each segment and every contributor in a content unit, under the constraint that a content unit can be allocated to no more than one segment. PyrScore applies WMIN (Sakai et al., 2003), a greedy algorithm, to solve this weighted set covering problem.

Both PyrScore and PEAK correlate well with manual pyramid scores, yet both methods have drawbacks. PyrScore does not construct a pyramid, so is not fully automated. ADW does random walks over WordNet, and ClauseIE generates a large number of low-quality triples; these steps result in high computation time. The PEAK pyramids are less interpretable because they contain many variants of the same CUs. PyrEval applies the same latent semantic methods used in PyrScore, and includes all the functionality of PEAK. It is faster than PEAK, and includes additional functions. To construct a pyramid, PyrEval relies on a new algorithm we developed (detailed in a separate publication), to produce a pyramid with a single version of each CU. It incorporates an improved version of PyrScore. Here we report on PyrEval's performance and present it for public use.

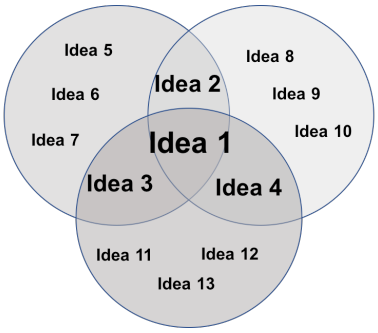

Figure 2: Venn diagram of summary content overlap

Segment from an automated summary:
Australia Sunday sent three Air Force C Hercules aircraft loaded with medical and food supplies on an urgent mission to help survivors of a devastating tsunami .

Matched content unit (Wt. = 4) from pyramid:
SUM1 Three Australian Cs were bringing food , medical supplies and personnel and a mobile hospital .
SUM2 Australia will transport relief supplies and provide a mobile hospital and medical personnel .
SUM3 Australia is sending medical supplies and food and is expecting to set up a mobile hospital .
SUM4 Australia also was sending a mobile hospital and doctors .

Figure 3: Example of a segment from D1004A automated summary and the matched content unit from D1004A PyrEval pyramid.

3. Pyramid Content Evaluation

The pyramid method was developed to capture the observation that high quality summaries from different individuals of the same source texts will express overlapping content, but will have even more content that is unique to each summary. This makes it challenging to define the gold standard content. Figure 2 illustrates the typical case by means of a Venn diagram to show that, given n reference summaries, among the set of all distinct ideas from the n summaries, relatively few ideas will occur in all n, and for each decreasing cardinality from n to 1, there will be a larger number of such ideas. Here one idea occurs in three summaries, three occur in two, and nine occur in only one. The font size of the schematically represented ideas in the figure reflects their importance, with the highest importance going to ideas that are expressed in all the reference summaries. Importance represents which ideas should be included in a new summary to adhere to the distribution of ideas found in the reference summaries.

The pyramid method defines content units (CUs) to correspond largely to clauses, and asks annotators to select all the phrases from distinct summaries that express the same content as a single CU (cf. Figure 1). The DUCView annotation tool developed for pyramid annotation elicits a label that the annotator writes. The CU consists of the label, and clauses (or sometimes phrases) selected from different reference summaries (e.g., D1, E3 and H4) that express the

Step	Description	Size	Runtime (sec)
1	Preprocessing with Stanford CoreNLP	26 summaries (312 sent; 6K words)	73.40
2	Sentence Decomposing and WTMF	same as above	208.68
3	Building Pyramid from Model Summaries	4 (human) summaries	4.35
4	Scoring Target Summaries by Pyramid	22 (machine) summaries	14.17

Table 1: Steps and times for PyrEval on one DUC topic(D0608).

same content. The weight is automatically assigned as the number of phrases. Each summary can contribute no more than once to the same content unit. The instructions require annotators to select, as far as possible, all the text from the reference summaries. The final pyramid consists of all the CUs identified by the annotator, with weights from 1 to N. To annotate a target summary against a pyramid model, the same annotation tool can be used to match phrases in the target summary to CUs in the pyramid. Figure 3 shows an example of scoring a segment from a target summary with the pyramid built by PyrEval. The sum of the weights of the matched CUs from the target summary is normalized in different ways to produce a score. Previous work has found this method to produce very stable and reliable scores (Nenkova et al., 2007), and to have very good inter-annotator agreement (Passonneau, 2010). Despite the labor intensiveness of manual annotation, pyramid has been applied in much work on summarization. We developed PyrEval to facilitate more widespread usage.

4. PyrEval

4.1. System Overview

Like PEAK, PyrEval implements pyramid construction and automated scoring. In contrast to PEAK, it can also score target summaries against a manually annotated pyramid that has been produced with the DUCView annotation tool. DUCView was developed for pyramid annotation, and has been used for all the NIST manual pyramid evaluations.

Two initial procedures for processing input texts are: Step 1-segmentation of sentences into clause-like units by

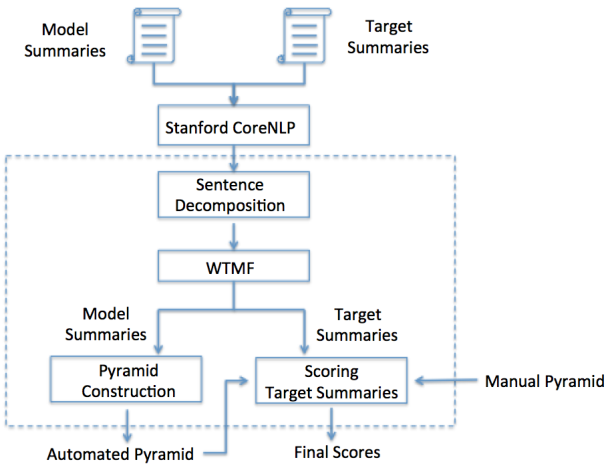

Figure 4: System Flow of PyrEval

The death toll, mostly children and old people, has reached 9 but is expected to rise and there are thousands of injured and homeless, with no food or water.

```
(ROOT (S
 (S
  (NP (NP (DT The) (NN death) (NN toll-3)) (, ,) (NP (NP (RB mostly)
  (NNS children)) (CC and) (NP (JJ old) (NNS people))) (, ,)) (VP (VP
  (VBZ has ) (VP (VBN reached-12)))
  (CC but) (VP (VBZ is) (VP (VBN expected -15) (S (VP (TO to) (VP (VB
  rise))))))))
  (CC and) (S
   (NP (EX there-19)) (VP (VBP are-20) (NP (NP (NNS thousands-21))
   (PP (IN of) (NP (JJ injured) (CC and) (JJ homeless)))) (, ,) (PP (IN
   with) (NP (DT no) (NN food) (CC or) (NN water)))))
 (. .)))
```

Figure 5: A sentence from D1004A and its constituency parse output from Stanford CoreNLP. The subtrees dominated by the nodes marked with bold font are the structures extracted by the decomposition parser.

PyrEval's sentence decomposition parser; Step 2-latent semantic representation via WTMF (see first two interior boxes in Figure 4). Then pairwise cosine similarities are computed between clause-like units from different model summaries, or clause-like units from target summaries and model CUs. Step 3 is Pyramid Construction, the second new component of PyrEval. Step 4 is PyrEval scoring; it is similar to PyrScore, but uses clause-like units rather than ngrams. Table 1 shows an example of running times for these four steps on one of the development sets from DUC06.

PyrEval was developed and tested on English. To apply it with other languages would require replacing the preprocessing steps that segment sentences into distinct clauses, and that produce low-dimensional vectors for their semantic representation. In principle, the current code package could easily be extended to substitute a sentence segmenter designed for a distinct language, and a pretrained matrix of words by sentences.

The installation for PyrEval requires Stanford CoreNLP (Manning et al., 2014), perl, python2.7 (or anaconda) with packages NLTK, statistics, beautifulsoup, networkx, sklearn and numpy. All the experiments reported here were conducted on an ubuntu machine, with 4 Intel i5-6600 CPUs.

4.1.1. Sentence Decomposition

PyrEval relies on our sentence decomposition parser to extract clause-like units (segments) from sentences. Each set of semantic units yielded by a sentence is a *segmentation*, and there will be at least one segmentation per sentence. The sentence decomposition parser takes results from the Stanford CoreNLP constituency parser and dependency parser as input. For each sentence, the decomposi-

root (ROOT-0 , reached-12), det (toll-3 , The-1) , compound (toll-3 , death-2), **nsubj (reached-12 , toll-3), nsubjpass (expected-15 , toll-3)** (extra), advmod (children-6 , mostly-5), appos (toll-3 , children-6), cc (children-6 , and-7), amod (people-9 , old-8), appos (toll-3 , people-9) (extra), conj:and (children-6 , people-9), aux (reached-12 , has-11), cc (reached-12 , but-13), auxpass (expected-15 , is-14), conj:but (reached-12 , expected-15), mark (rise-17 , to-16), xcomp (expected-15 , rise-17), cc (reached-12 , and-18), **expl (are-20 , there-19)**,conj:but (reached-12 , are-20), **nsubj (are-20 , thousands-21)**, case (injured-23 , of-22), nmod:of (thousands-21 , injured-23), cc (injured-23 , and-24), nmod:of (thousands-21 , homeless-25) (extra), conj:and (injured-23 , homeless-25), case (food-29 , with-27), neg (food-29 , no-28),nmod:with (are-20 , food-29),cc (food-29 , or-30), nmod:with (are-20 , water-31) (extra), conj:or (food-29 , water-31)

Figure 6: Enhanced dependency parser output from Stanford CoreNLP for the sentence in Figure 5.The bold fonts are the dependency relations extracted by our decomposition parser.

Sentence Decomposition Example

	Segmentation 1
Segment 1	The death toll, mostly children and old people has reached 9 but is expected to rise
Segment 2	and there are thousands of injured and homeless with no food or water .
	Segmentation 2
Segment 1	toll The death, mostly children and old people has reached 9 but
Segment 2	toll is expected to rise and there are thousands of of injured and homeless, with no food or water.

Figure 7: Sentence decomposition parser output for sentence shown in Figure 5

tion parser extracts verb phrases and clauses from the constituency parses, and relation arcs from dependency parses, then associates verb phrases with subjects. Unconsumed words are attached to segments based on their linear position relative to words in segments. Each segmentation contains one or more segments (see Figure 7), and covers all words in a sentence. Every segment is then represented as a WMTF vector.

4.1.2. Pyramid Construction

In this paper, we give a high level description of the algorithm for constructing the pyramid. Each candidate content unit is represented as an undirected, complete graph G where vertices are vectors of segments from different summaries, and edges are cosine similarities (See Figure 3). To construct CU graphs, pairwise similarities between segments from different summaries are computed, and edges are added if every pair of vertices v_i, v_j has a cosine similarity above a fixed threshold t. Semantic coherence of a CU graph is measured as the Average Similarity (AS) of all segment pairs. Given G with n vertices, we define k as the number of edges and calculate AS as:

$$AS = \frac{\sum_{u,v \in G, u \neq v} similarity(u, v)}{k} \quad (1)$$

CU weight produces a partition over all CUs; that is, all CUs of the same weight are one equivalence class. The sizes of the n equivalence classes for n reference summaries are observed to have a power law distribution inversely related to the weights (Nenkova et al., 2007); that is, the smallest class is the one with the highest weight. We developed a greedy algorithm to allocate segments to CUs that achieves the maximum average of the AS values for each equivalence class, and that adheres to a power law distribution.

4.1.3. Automated Scoring

PyrEval can score target summaries using a manually or automatically constructed pyramid. As with pyramid construction, sentences in target summaries are processed by the Stanford CoreNLP tools and sentence decomposition parser to produce candidate segmentations for each sentence, then WTMF is used to generate their vector representations. The scoring function is the one used in PyrScore. The quantitative output includes a raw score (the sum of matched CU weights), and several ways to normalize. The user can choose to output a qualitative analysis as well.

5. Experiments

We first tested PyrEval on single-document summaries from community college students that PEAK and PyrScore were tested on (Passonneau et al., 2017). PyrEval has four parameters: two that control the size of the equivalence classes corresponding to CUs of each weight (a, b), and two similarity thresholds (for CU construction, t_1; for scoring, t_2). In each set of reference summaries, cosine similarities are computed for all pairs of segments from different summaries. Values are distributed differently for different sets of reference summaries, so we define t_1 as a percentile over the range of observed cosine values for a given set of reference summaries, and conduct grid search over a range of percentile values, as detailed below. For t_2 we use 0.50 cosine similarity, as in previous work (Passonneau et al., 2017). The Pearson correlation with manual pyramid scores on 20 student summaries reached 89%, a 10% improvement over the previous results.

When we turned to DUC and TAC datasets (see above), consisting of extractive summarizers on newswire with model summaries written by NIST assessors, new challenges arose. The decomposition parser for all results presented here was extended to handle the more complex syntax found in newswire, relative to the student summaries.

We performed grid search for DUC06 dev consisting of four of the twenty topics, and selected the best four sets of parameters. For each next year, we tested these same four parameter sets on a randomly selected dev set of size four, and found the best performance for $a = 100$, $b = 2.5$, $t_1 = 83\%$; these values yielded the smallest standard deviations for average correlations with the manual pyramids. The value of the a parameter is apparently sensitive to the number of reference summaries. We tested PyrEval on the same student data reported in (Passonneau et al., 2017), which has five model summaries instead of the four that come with the NIST datasets. When we did grid search on a development set of the student data, we got better performance for $a \in [125, 250]$.

Table 2 presents results on five years of DUC/TAC. In each year, testing was performed on all data apart from the devel-

Dataset	Size	Avg	Med	Min	Max	StdDev
DUC06	16	41.47	46.62	-2.31	76.34	24.42
DUC07	19	44.66	39.63	-3.27	89.18	27.05
TAC08	96	42.26	43.29	-13.93	79.09	18.92
TAC09	88	55.14	57.36	15.64	84.54	16.67
TAC10	92	51.95	53.33	10.16	83.88	16.27

Table 2: Pearson correlations (Pearson's $r \times 100$) with manual pyramid scores from DUC and TAC.

opment set. Because the semantic vectors are constructed from pretrained data, there is no training set. As shown, the trend over time is for PyrEval's average correlation to be higher, and for the standard deviation to be lower. This could result simply from the much larger datasets as of 2008, or that PyrEval performs better on data from summarizers whose performance is better. Conroy & Dang (2008) report that the top ROUGE scores increase from about 0.27 in 2005 to about 0.34 in 2007. The maximum correlations for an individual topic in one year can be as high as 0.89. The minimum values tend to be many standard deviations below the mean, suggesting that these are outliers, and that the median is a better summary statistic. Apart from DUC07, the medians are higher than the means.

6. Discussion and Conclusion

The results in Table 2 show that PyrEval can achieve good average correlations of up to 0.55 (TAC10), and much higher (e.g., 0.89) on individual summarization tasks. As discussed in the next paragraph, we have identified a number of needed improvements we plan to address.

Inspection of the pyramid for one of the DUC06 dev topics (D0608) that had lower performance illustrates one of the two issues our future work will address, both of which are classic problems in NLP: noun-noun compounds (including Named Entities) and conjunction. As shown in CU 1 of Figure 8, there are noun-noun compounds and NEs in all four segments (e.g., "booster seat", "back seat", "crash test", "sport utility model"; "MercedesBenz", "BodySmart"). The vector representations treat the compounds as sequences of individual words, which does not capture the correct semantics. Identification of multi-word expressions prior to generating the phrase vectors might correct this. We have also observed that conjunctions, which typically introduce a great deal of syntactic ambiguity, are often parsed incorrectly. This can lead to incoherent output from the decomposition parser. Possible methods to address this include neural net dependency parsing, as in (Ficler and Goldberg, 2017), or clause-chart parsing (Kríž and Hladká, 2016).

Figure 8 also shows a CU produced by PyrEval that captures the same content as the manual content unit shown in Figure 1. The PyrEval CU has no label, but is otherwise similar to the manual CU: it has the same weight, and the contributing clauses come from the same sentences of the same summaries. The phrases selected by the human annotator for this CU are subphrases of the clauses that PyrEval groups together. Future work could rely on improved parsing to identify all distinct propositions, not just tensed clauses.

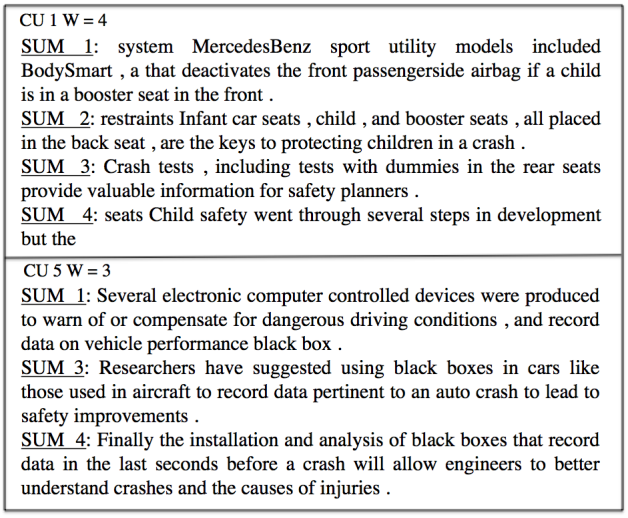

Figure 8: Examples of an incoherent CU (CU 1) and a coherent CU (CU 5) from DUC0608.

In sum, PyrEval is a fast, complete, multi-use tool. In tests on summaries from extractive, multi-document summarizers, it achieves good average correlations with manual pyramid evaluations on five years of DUC/TAC .

7. References

Bing, L., Li, P., Liao, Y., Lam, W., Guo, W., and Passonneau, R. (2015). Abstractive multi-document summarization via phrase selection and merging. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1587–1597, Beijing.

Conroy, J. M. and Dang, H. T. (2008). Mind the gap: Dangers of divorcing evaluations of summary content from linguistic quality. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1 (COLING '08)*, pages 145–152, Stroudsburg, PA, USA. Association for Computational Linguistics.

Del Corro, L. and Gemulla, R. (2013). ClausIE: clause-based open information extraction. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 355–366. ACM.

Durrett, G., Berg-Kirkpatrick, T., and Klein, D. (2016). Learning-based single-document summarization with compression and anaphoricity constraints. In *Proceedings of the 54th Annual Meeting of the Association for*

Computational Linguistics (Volume 1: Long Papers), pages 1998–2008, Berlin.

Ficler, J. and Goldberg, Y. (2017). Improving a strong neural parser with conjunction-specific features. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL): Volume 2, Short Papers*, pages 343–348, Valencia, Spain, April. Association for Computational Linguistics.

Guo, W. and Diab, M. (2012). Modeling sentences in the latent space. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 864–872.

Ji, H., Grishman, R., Dang, H. T., Griffitt, K., and Ellis, J. (2010). Overview of the TAC 2010 knowledge base population track. In *Third Text Analysis Conference (TAC 2010)*, volume 3, pages 3–28.

Kríž, V. and Hladká, B. (2016). Improving dependency parsing using sentence clause charts. In *Proceedings of the ACL 2016 Student Research Workshop*, pages 86–92.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics (NRL)*, 2(1-2):83–97.

Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In Stan Szpakowicz Marie-Francine Moens, editor, *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8, pages 74–81. Barcelona, Spain.

Liu, F., Flanigan, J., Thomson, S., Sadeh, N., and Smith, N. A. (2015). Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086.

Louis, A. and Nenkova, A. (2009). Automatically evaluating content selection in summarization without human models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 306–314. Association for Computational Linguistics.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., and McClosky, D. (2014). The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Nema, P., Khapra, M. M., Laha, A., and Ravindran, B. (2017). Diversity driven attention model for query-based abstractive summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1063–1072, Vancouver, Canada, July.

Nenkova, A. and Passonneau, R. (2004). Evaluating content selection in summarization: The pyramid method. In Susan Dumais, et al., editors, *Proceedings of Human Language Technologies and the North American Association of Computational Linguistics (HLT-NAACL 2004)*, pages 145–152, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Nenkova, A., Passonneau, R., and McKeown, K. (2007). The pyramid method: Incorporating human content selection variation in summarization evaluation. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(2):1–23. Article 4.

Passonneau, R. J., McKeown, K., Sigelman, S., and Goodkind, A. (2006). Applying the pyramid method in the 2006 document understanding conference. In *Proceedings of the Document Understanding Conference (DUC 06)*.

Passonneau, R. J., Chen, E., Guo, W., and Perin, D. (2013). Automated pyramid scoring of summaries using distributional semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 143–147.

Passonneau, R. J., Poddar, A., Gite, G., Krivokapic, A., Yang, Q., and Perin, D. (2017). Wise crowd content assessment and educational rubrics. *International Journal of Artificial Intelligence in Education*, pages 1–27.

Passonneau, R. J. (2010). Formal and functional assessment of the pyramid method for summary content evaluation. *Natural Language Engineering*, 16:107–131.

Peyrard, M. and Eckle-Kohler, J. (2017). Supervised learning of automatic pyramid for optimization-based multi-document summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1084–1094.

Pilehvar, M. T., Jurgens, D., and Navigli, R. (2013). Align, disambiguate and walk: A unified approach for measuring semantic similarity. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1341–1351.

Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, September.

Sakai, S., Togasaki, M., and Yamazaki, K. (2003). A note on greedy algorithms for the maximum weighted independent set problem. *Discrete Applied Mathematics*, 126(2):313–322.

Varma, V., Pingali, P., Katragadda, R., Krishna, S., Ganesh, S., Sarvabhotla, K., Garapati, H., Gopisetty, H., Reddy, V. B., Reddy, K., et al. (2009). IIIT hyderabad at TAC 2009. In *Second Text Analysis Conference (TAC 2009)*.

Yang, Q., Passonneau, R. J., and de Melo, G. (2016). PEAK: Pyramid evaluation via automated knowledge extraction. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2673–2680.