Using the Omega Index for Evaluating Abstractive Community Detection

Gabriel Murray
Computer Information Systems
University of the Fraser Valley
gabriel.murray@ufv.ca

Giuseppe Carenini
Computer Science
University of British Columbia
carenini@cs.ubc.ca

Raymond Ng
Computer Science
University of British Columbia
rng@cs.ubc.ca

Abstract

Numerous NLP tasks rely on clustering or community detection algorithms. For many of these tasks, the solutions are disjoint, and the relevant evaluation metrics assume non-overlapping clusters. In contrast, the relatively recent task of *abstractive community detection* (ACD) results in overlapping clusters of sentences. ACD is a sub-task of an abstractive summarization system and represents a two-step process. In the first step, we classify sentence pairs according to whether the sentences should be realized by a common abstractive sentence. This results in an undirected graph with sentences as nodes and predicted abstractive links as edges. The second step is to identify communities within the graph, where each community corresponds to an abstractive sentence to be generated. In this paper, we describe how the Omega Index, a metric for comparing non-disjoint clustering solutions, can be used as a summarization evaluation metric for this task. We use the Omega Index to compare and contrast several community detection algorithms.

1 Introduction

Automatic summarization has long been proposed as a helpful tool for managing the massive amounts of language data in our modern lives (Luhn, 1958; Edmundson, 1969; Teufel and Moens, 1997; Carbonell and Goldstein, 1998; Radev et al., 2001). Most summarization systems are *extractive*, meaning that a subset of sentences from an input document forms a summary of the whole. Particular significance may be attached to the chosen sentences, e.g. that they are relevant to a provided query, generally important for understanding the overall document, or represent a particular phenomenon such as action items from a meeting. In any case, extraction consists of binary classification of candidate sentences, plus post-processing steps such as sentence ranking and compression. In contrast, recent work attempts to replicate the *abstractive* nature of human-authored summaries, wherein new sentences are generated that describe the input document from a higher-level perspective. While some abstractive summary sentences are very similar to individual sentences from the document, others are created by synthesizing multiple document sentences into a novel abstract sentence. In this paper, we address a component of this latter task, namely identifying which sentences from the source documents should be combined in generated abstract sentences. We call this task *abstractive community detection* (ACD), and apply the task to a publicly available meeting dataset.

Herein we focus on describing how the Omega Index (Collins and Dent, 1988), a metric for comparing non-disjoint clustering solutions, can be used as a summarization evaluation metric for the ACD task. Metrics such as the Rand Index (Rand, 1971) are insufficient since they are intended only for disjoint clusters.

ACD itself is carried out in two steps. First, we classify sentence pairs according to whether they should be realized by a common abstractive sentence. For this step, we use supervised machine learning that exploits human-annotated links between abstracts and extracts for a given document. This results in an undirected graph with nodes representing sentences and edges representing predicted abstractive links. Second, we identify communities within the graph, where each community corresponds to an abstractive sentence to be generated. We experiment with several divisive community de-

Proceedings of the Workshop on Evaluation Metrics and System Comparison for Automatic Summarization, pages 10–18,
Montréal, Canada, June 3-8, 2012.

tection algorithms, and highlight the importance of selecting an algorithm that allows overlapping communities, owing to the fact that a document sentence can be expressed by, and linked to, more than one abstract summary sentence in the gold-standard.

The structure of the paper is as follow. In Section 2, we compare and contrast ACD with other relevant tasks such as extractive summarization and topic clustering. In Sections 3-4, we describe the two ACD steps before we can fully discuss evaluation methods. Section 5 describes the experimental setup and corpora used, including a description of the abstractive and extractive summary annotations and the links between them. In Section 6, we give a detailed description of the Omega Index and explain how it differs from the more common Rand Index. In Sections 7-8 we present results and draw conclusions.

2 Related Work

The ACD task differs from more common extractive summarization (Mani, 2001a; Jurafsky and Martin, 2008). Whereas extraction involves simply classifying sentences as important or not, ACD is a sub-task of *abstractive* summarization wherein document sentences are grouped according to whether they can be jointly realized by a common abstractive sentence. The first step of ACD, where we predict links between sentence pairs, can be seen to encompass extraction since the link is via an as-yet-ungenerated abstract sentence, i.e. each linked sentence is considered summary-worthy. However, the second step moves away from extraction by clustering the linked sentences from the document in order to generate abstract summary sentences.

ACD also differs from topic clustering (Malioutov and Barzilay, 2006; Joty et al., 2010), though there are superficial similarities. A first observation is that topic links and abstract links are genuinely different phenomena, though sometimes related. A single abstract sentence can reference more than one topic, e.g. *They talked about the interface design and the budget report*, and a single topic can be referenced in numerous abstract sentences. From a practical standpoint, in our work on ACD we cannot use many of the methods and evaluation metrics designed for topic clustering, due to the fact that a document sentence can belong to more than one abstract sentence. This leads to overlapping communities, whereas most work on topic clustering has focused primarily on disjoint communities where a sentence belongs to a single topic. In Section 4, we discuss community detection algorithms and evaluation metrics that allow overlapping communities.

Work on detecting *adjacency pairs* (Shriberg et al., 2004; Galley et al., 2004) also involves classifying sentence pairs as being somehow related. For example, if sentence B directly follows sentence A, we might determine that they have a relationship such as question-answer or request-accept. In contrast, with ACD there is no requirement that sentence pairs be adjacent or even in proximity to one another, nor must they be in a rhetorical relation.

Work on *sentence fusion* (Barzilay and McKeown, 2005) identifies sentences containing similar or repeated information and combines them into new sentences. In contrast, in our task sentences need not contain repeated information in order to be linked. For example, two sentences could be linked to a common abstract sentence due to a more complex rhetorical relationship such as proposal-reject or question-answer.

ACD is a more general problem that may incorporate elements of topic clustering, adjacency pair detection and other sentence clustering or pairing tasks. Here we try to directly learn the abstractive sentence links using lower-level features such as shared n-grams and cosine similarity, as described in Section 3, but in future work we will model higher-level features of topics and rhetorical structure.

3 Step 1: Building a Sentence Pair Graph

In order to describe the use of the Omega Index for the ACD task, we must first introduce the ACD task in some detail. The first step in ACD is to determine which sentence pairs are linked. If two sentences are linked, it means they can be at least partly realized in the abstract summary by a common sentence. A document sentence may "belong" to more than one abstract sentence. We take a supervised classification approach to this problem, training on a dataset containing explicit links between extract sentences and abstract sentences. The corpus and relevant annotation are described in detail in Section 5. For

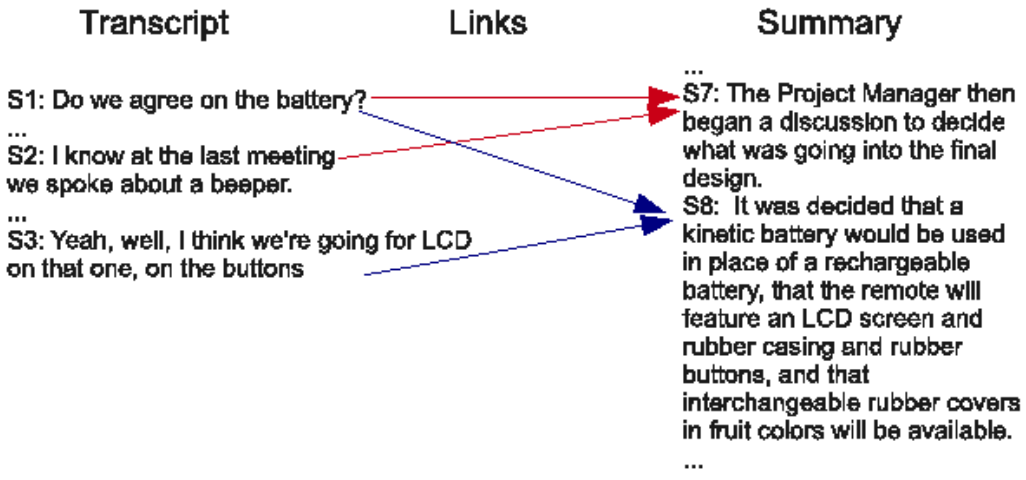

Figure 1: Linked Sentences

our gold-standard data, a sentence pair is considered linked if both sentences are linked to a common abstract sentence and not-linked otherwise.

Figure 1 shows an example snippet of linked sentences from our corpus. The first and second sentences are linked via one abstract sentence while the first and third sentences are linked via a different abstract sentence. While it is not shown in this example, note that two sentences can also be linked via more than one abstract sentence.

We take a supervised machine learning approach toward predicting whether a sentence pair is linked. For each pair, we extract features that can be classed as follows:

- **Structural:** The *intervening number* of sentences, the *document position* as indicated by the midpoint of the two sentences, the *combined length* and the *difference in length* between the two sentences, and whether the two sentences share the *same speaker*.
- **Linguistic:** The number of *shared bigrams*, *shared part-of-speech tags*, the *sum and average of tf.idf weights*, and the *cosine similarity* of the sentence vectors.

We run the trained classifier over sentence pairs, predicting abstractive links between sentences in the document. This results in an unweighted, undirected graph where nodes represent sentences and edges

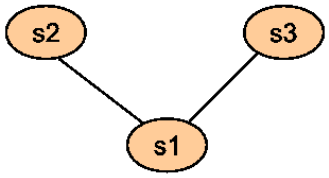

Figure 2: Graph with Sentence Nodes

represent an abstractive link. Continuing with the conversation snippet from Figure 1, we would end up with a graph like Figure 2. This very simple example of a graph shows that there are abstractive links predicted between sentences s1 and s2 and between sentences s1 and s3. There is no direct link predicted between sentences s2 and s3. However, it is possible for two sentences with no predicted link between them to wind up in the same abstractive community after running a community detection algorithm on the graph. We discuss this community detection step in the following section.

4 Step 2: Discovering Abstractive Sentence Communities

In the first step of ACD, we predicted whether pairs of sentences can be at least partly realized by a common abstractive sentence. We then want to identify *communities* or clusters within the graph. Each of these communities will correspond to an abstractive

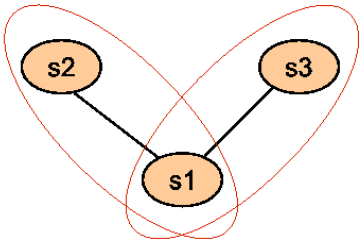

Figure 3: Overlapping Communities in Graph

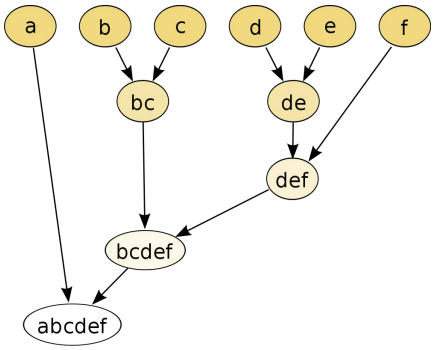

Figure 4: Community Dendrogram

sentence that we will generate. Continuing with our simple example, Figure 3 shows two communities that have been identified in the graph. Note that the communities are overlapping, as each contains sentence s1; we would generate one abstractive sentence describing sentences s1 and s2 and another describing sentences s1 and s3. We will return to this critical issue of overlapping communities shortly.

The task of identifying communities in networks or graphs has received considerable attention (Porter et al., 2009). The Girvan-Newman algorithm (Girvan and Newman, 2002) is a popular community detection method based on a measure of *betweenness*. The betweenness score for an edge is the number of shortest paths between pairs of nodes in the graph that run along that edge. An edge with a high betweenness score is likely to be between two communities and is therefore a good candidate for removal, as the goal is to break the initial graph into distinct communities. The Girvan-Newman algorithm proceeds as follows:

1. Calculate the betweenness of each edge in the graph.
2. Remove the edge with the highest betweenness.
3. For any edge affected by Step 2, recalculate betweenness.
4. Repeat steps 2 and 3 until no edges remain

In this way we proceed from the full graph with all edges intact to the point where no edges remain and each node is in its own community. The intermediate steps can be visualized by the resulting dendrogram, such as seen in Figure 4 [1].

The top row, the "leaves" of the dendrogram, represents the individual nodes in the graph. The rest of the dendrogram shows how these nodes are situated in nested communities, e.g. *b* and *c* form a community *bc* that combines with *def* to form *bcdef*. In our case, where nodes are sentences, the dendrogram shows us how sentences combine into nested communities. This can be useful for generating abstracts of different granularities, e.g. we could describe *bcdef* in one sentence or generate two sentences to separately describe *bc* and *def*.

The drawback of Girvan-Newman for our purposes is that it does not allow overlapping communities, and we know that our human-annotated data contain overlaps. Note from Figure 4 that all communities decompose into disjoint nested communities, such as *bcdef* being comprised of *bc* and *def*, not *bc* and *bdef* or some other overlapping case. We therefore hypothesize that Girvan-Newman in its traditional form is not sufficient for our current research. For the same reason, recent graph-based approaches to topic clustering (Malioutov and Barzilay, 2006; Joty et al., 2010) are not directly applicable here.

It is only in recent years that much attention has been paid to the problem of overlapping (or non-disjoint) communities. Here we consider two recent modifications to the Girvan-Newman algorithm that allow for overlaps. The CONGA algorithm (Gregory, 2007) extends Girvan-Newman so that instead of removing an edge on each iteration, we either remove an edge or copy a node. When a node is copied, an overlap is created. Nodes are associated with a betweenness score (called the *split betweenness*) derived from the edge betweenness scores, and at each step we either remove the edge with the highest betweenness score or copy the node with the

[1] Image Source: Wikimedia Commons (Mhbrugman)

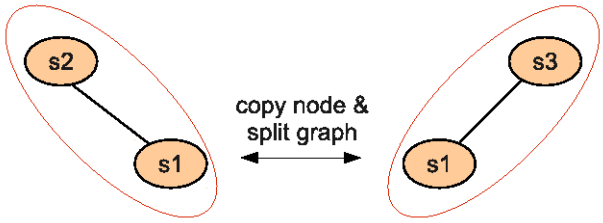

Figure 5: CONGA algorithm

highest split betweenness, if it is greater. The edge and node betweenness scores are then recalculated. In such a manner we can detect overlapping communities. Figure 5 shows the CONGA copying and splitting operations applied to our simple example, so that sentence s1 now exists in two communities.

The CONGO algorithm (Gregory, 2008) is an approximation of CONGA that is more efficient for large graphs. Girvan-Newman (and hence CONGA) are not feasible algorithms for very large graphs, due to the number of repeated betweenness calculations. CONGO addresses this problem by using *local betweenness* scores. Instead of calculating betweenness using the shortest paths of every pair of nodes in the graph, only nodes within a given horizon h of an edge are considered. When $h = \infty$ then CONGO and CONGA are identical. Gregory (Gregory, 2008) found good results using $h = 2$ or $h = 3$ on a variety of datasets including blog networks; here we experiment $h = 2$.

For the community detection step of our system, we run both CONGA and CONGO on our graphs and compare our results with the Girvan-Newman algorithm. For all community detection methods, as well as human annotations, any sentences that are not linked to at least one other sentence in Step 1 are assigned to their own singleton communities. Also, the algorithms we are evaluating are hierarchical (see Figure 4), and we evaluate at $n = 18$ clusters, since that is the average number of sentences per abstractive meeting summary in the training set.

5 Experimental Setup

In this section we describe the dataset used, including relevant annotations, as well as the statistical classifiers used for Step 1.

5.1 AMI Corpus

For these experiments we use the AMI meeting corpus (Carletta, 2006), specifically, the subset of scenario meetings where participants play roles within a fictional company. For each meeting, an annotator first authors an abstractive summary. Multiple annotators then create extractive summaries by linking sentences from the meeting transcript to sentences within the abstract. This generates a many-to-many mapping between transcript sentences and abstract sentences, so that a given transcript sentence can relate to more than one abstract sentence and vice-verse. A sample of this extractive-abstractive linking was shown in Figure 1.

It is known that inter-annotator agreement can be quite low for the summarization task (Mani et al., 1999; Mani, 2001b), and this is the case with the AMI extractive summarization codings. The average κ score is 0.45.

In these experiments, we use only human-authored transcripts and plan to use speech recognition transcripts in the future. Note that our overall approach is not specific to conversations or to speech data. Step 2 is completely general, while Step 1 uses a single *same-speaker* feature that is specific to conversations. That feature can be dropped to make our approach completely general (or, equivalently, that binary feature can be thought of as always 1 when applied to monologic text).

5.2 Classifiers

For Step 1, predicting abstractive links between sentences, we train a logistic regression classifier using the liblinear toolkit[2]. The training set consists of 98 meetings and there are nearly one million sentence pair instances since we consider every pairing of sentences within a meeting. The test set consists of 20 meetings on which we perform our evaluation.

6 Evaluation Metrics

In this section, we present our evaluation metrics for the two steps of the task.

6.1 Step 1 Evaluation: PRF and AUROC

For evaluating Step 1, predicting abstractive sentence links, we present both precision/recall/f-score

[2]http://www.csie.ntu.edu.tw/ cjlin/liblinear/

as well as the area under the receiver operator characteristic curve (AUROC). While the former scores evaluate the classifier at a particular posterior probability threshold, the AUROC evaluates the classifier more generally by comparing the true-positive and false-positive rates at varying probability thresholds.

6.2 Step 2 Evaluation: The Omega Index

For evaluating Step 2, ACD, we employ a metric called the Omega Index which is designed for comparing disjoint clustering solutions. To describe and motivate our use of this metric, it is necessary to describe previous metrics upon which the Omega Index improves. The Rand Index (Rand, 1971) is a way of comparing disjoint clustering solutions that is based on pairs of the objects being clustered. Two solutions are said to agree on a pair of objects if they each put both objects into the same cluster or each into different clusters. The Rand Index can then be formalized as

$$(a + d)/N$$

where N is the number of pairs of objects, a is the number of times the solutions agree on putting a pair in the same cluster and d is the number of times the solutions agree on putting a pair in different clusters. That is, the Rand Index is the number of pairs that are agreed on by the two solutions divided by the total number of pairs. The Rand Index is insufficient for overlapping solutions because pairs of objects can exist together in more than one community. In those cases, two solutions might agree on the occurrence of a pair of objects in one community but disagree on the occurrence of that pair in another community. The Rand Index cannot capture that distinction.

An improvement to the Rand Index is the Adjusted Rand Index (Hubert and Arabie, 1985) which adjusts the level of agreement according to the expected amount of agreement based on chance. However, the Adjusted Rand Index also cannot account for disjoint solutions.

The Omega Index (Collins and Dent, 1988) builds on both the Rand Index and Adjusted Rand Index by accounting for disjoint solutions and correcting for chance agreement. The Omega Index considers the *number* of clusters in which a pair of objects is together. The observed agreement between solutions is calculated by

$$Obs(s1, s2) = \sum_{j=0}^{min(J,K)} A_j/N$$

where J and K represent the maximum number of clusters in which any pair of objects appears together in solutions 1 and 2, respectively, A_j is the number of the pairs agreed by both solutions to be assigned to number of clusters j, and N is again the number of pairs of objects. That is, the observed agreement is the proportion of pairs classified the same way by the two solutions. The expected agreement is given by:

$$Exp(s1, s2) = \sum_{j=0}^{min(J,K)} N_{j_1} N_{j_2}/N^2$$

where N_{j_1} is the total number of pairs assigned to number of clusters j in solution 1, and N_{j_2} is the total number of pairs assigned to number of clusters j in solution 2. The Omega Index is then calculated as

$$Omega(s1, s2) = \frac{Obs(s1, s2) - Exp(s1, s2)}{1 - Exp(s1, s2)}$$

The numerator is the observed agreement adjusted by expected agreement, while the denominator is maximum possible agreement adjusted by expected agreement. The highest possible score of 1 indicates that two solutions perfectly agree on how each pair of objects is clustered. With the Omega Index, we can now evaluate the overlapping solutions discovered by our community detection algorithms.[3]

7 Results

In this section we present the results for both steps of ACD. Because the Omega Index is not used for evaluating Step 1, we keep that discussion brief.

7.1 Step 1 Results: Predicting Abstractive Sentence Links

For the task of predicting abstractive links within sentence pairs, the resulting graphs have an average of 133 nodes and 1730 edges, though this varies

[3]Software for calculating the Omega Index will be released upon publication of this paper.

System	Prec.	Rec.	F-Score	AUROC
Lower-Bound	0.18	1	0.30	0.50
Message Links	0.30	0.03	0.05	-
Abstractive Links	0.62	0.54	**0.54**	**0.89**

Table 1: P/R/F and AUROCs for Link Prediction

widely depending on meeting length (from 37 nodes and 61 edges for one short meeting to 224 edges and 5946 edges for a very long meeting). In comparison, the gold-standard graphs have an average of 113 nodes and 1360 edges. The gold-standards similarly show huge variation in graph size depending on meeting length.

Table 1 reports both the precision/recall/f-scores as well as the AUROC metrics. We compare our supervised classifier (labeled "Abstractive Links") with a lower-bound where all instances are predicted as positive, leading to perfect recall and low precision. Our system scores moderately well on both precision and recall, with an average f-score of 0.54. The AUROC for the abstractive link classifier is 0.89.

It is difficult to compare with previous work since, to our knowledge, nobody has previously modeled these extractive-abstractive mappings between document sentences and associated abstracts. We can compare with the results of Murray et al. (2010), however, who linked sentences by aggregating them into *messages*. In that work, each message is comprised of sentences that share a dialogue act type (e.g. an action item) and mention at least one common entity (e.g. *remote control*). Similar to our work, sentences can belong to more than one message. We assess how well their message-based approach captures these abstractive links, reporting their precision/recall/f-scores for this task in Table 1, with their system labeled "Message Links". While their precision is above the lower-bound, the recall and f-score are extremely low. This demonstrates that their notion of message links does not capture the phenomenon of abstractive sentence linking.

7.2 Step 2 Results: Discovering Abstractive Communities

For the task of discovering abstractive communities in our sentence graphs, Table 2 reports the Omega Index for the CONGA, CONGO and Girvan-Newman algorithms. We also report the average Omega Index for the human annotators themselves, derived by comparing each pair of annotator solutions for each meeting.

It is not surprising that the Omega Index is low for the inter-annotator comparison; we reported previously that the κ score for the extractive summaries of this corpus is 0.45. That κ score indicates that there is high disagreement about which sentences are most important in a meeting. We should not be surprised then that there is further disagreement about how the sentences are linked to one another. What *is* surprising is that the automatic community detection algorithms achieve higher Omega Index scores than do the annotators. Note that the higher scores of the community detection algorithms relative to human agreement is *not* simply an artefact of identifying clustering solutions that have more overlap than human solutions, since even the disjoint Girvan-Newman solutions are higher than inter-annotator levels. One possible explanation is that the annotators are engaged in a fairly local task when they create extractive summaries; for each abstractive sentence, they are looking for a set of sentences from the document that relate to that abstract sentence, and because of high redundancy in the document the different annotators choose subsets of sentences that have little overlap but are still similar (Supporting this, we have found that we can train on annotator A's extractive codings and test on annotator B's and get good classification results even if A and B have a low κ score.). In contrast, the community detection algorithms are taking a more comprehensive, global approach by considering all predicted links between sentences (Step 1) and identifying the overlapping communities among them (Step 2).

When looking for differences between automatic and human community detection, we observed that the algorithms assigned more overlap to sentences

System	Omega
Girvan-Newman	0.254
CONGA	**0.263**
CONGO	0.241
Human	0.209

Table 2: Omega Index for Community Detection

System	Omega All Annots.	Omega 1 Annot.
Girvan-Newman	0.445	0.878
CONGA	**0.454**	**0.896**
CONGO	0.453	0.894

Table 3: Omega Index, Gold-Standard Graphs

than did the human annotators. For example, the CONGA algorithm assigned each sentence to an average of 1.1 communities while the human annotators assigned each to an average of 1.04 communities. Note that every sentence belongs to at least one community since unlinked sentences belong to their own singleton communities, and most sentences are unlinked, explaining why both scores are close to 1.

Comparing the algorithms themselves, we find that CONGA is better than both Girvan-Newman (marginally significant, $p = 0.07$) and CONGO ($p = 0.015$) according to paired t-test. We believe that the superiority of CONGA over Girvan-Newman points to the importance of allowing overlapping communities. And while CONGO is an efficient approximation of CONGA that can be useful for very large graphs where CONGA and Girvan-Newman cannot be applied, in these experiments the local betweenness used by CONGO leads to lower overall scores. Furthermore, our networks are small enough that both CONGA and Girvan-Newman are able to finish quickly and there is therefore no need to rely on CONGO.

Our Step 2 results are dependent on the quality of the Step 1 results. We therefore test how good our community detection results would be if we had gold-standard graphs rather than the imperfect output from Step 1. We report two sets of results. In the first case, we take an annotator's gold-standard sentence graph showing links between sentences and proceed to run our algorithms over that graph, comparing our community detection results with the communities detected by all annotators. In the second case, we again take an annotator's gold-standard graph and apply our algorithms, but then only compare our community detection results with the communities detected by the annotator who supplied the gold-standard graph. Table 3 shows both sets of results. We can see that the latter set contains much higher scores, again reflecting that annotators disagree with each other on this task.

Given gold-standard sentence graphs, CONGA and CONGO perform very similarly; the differences are negligible. Both are substantially better than the Girvan-Newman algorithm (all $p < 0.01$). This tells us that it is necessary to employ community detection algorithms that allow overlapping communities. These results also tell us that the CONGO algorithm is more sensitive to errors in the Step 1 output since it performed well using the gold-standard but worse than Girvan-Newman when using the automatically derived graphs.

8 Conclusion

After giving an overview of the ACD task and our approach to it, we described how the Omega Index can be used as a summarization evaluation metric for this task, and explained why other community detection metrics are insufficient. The Omega Index is suitable because it can account for overlapping clustering solutions, and corrects for chance agreement.

The main surprising result was that all of the community detection algorithms have higher Omega Index scores than the human-human Omega scores representing annotator agreement. We have offered one possibe explanation; namely, that while the human annotators have numerous similar candidate sentences from the document that each could be linked to a given abstract sentence, they may be satisfied to only link (and thus extract) a small representative handful, whereas the community detection algorithms work to find all extractive-abstractive links. We plan to further research this issue, and potentially derive other evaluation metrics that better account for this phenomenon.

References

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

J. Carbonell and J. Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proc. of ACM SIGIR Conference on Research and Development in Information Retrieval 1998, Melbourne, Australia*, pages 335–336.

J. Carletta. 2006. Unleashing the killer corpus: experiences in creating the multi-everything ami meeting corpus. In *Proc. of LREC 2006, Genoa, Italy*, pages 181–190.

L. Collins and C. Dent. 1988. Omega: A general formulation of the rand index of cluster recovery suitable for non-disjoint solutions. *Multivariate Behavioral Research*, 23:231–242.

H. P. Edmundson. 1969. New methods in automatic extracting. *J. ACM*, 16(2):264–285.

M. Galley, K. McKeown, J. Hirschberg, and E. Shriberg. 2004. Identifying agreement and disagreement in conversational speech: Use of bayesian networks to model pragmatic dependencies. In *Proc. of ACL 2004*.

M. Girvan and M.E.J. Newman. 2002. Community structure in social and biological networks. *Proc. of the National Academy of Sciences*, 99:7821–7826.

S. Gregory. 2007. An algorithm to find overlapping community structure in networks. In *Proc. of ECML/PKDD 2007, Warsaw, Poland*.

S. Gregory. 2008. A fast algorithm to find overlapping communities in networks. In *Proc. of ECML/PKDD 2008, Antwerp, Belgium*.

L. Hubert and P. Arabie. 1985. Comparing partitions. *Journal of Classification*, 2:193–218.

S. Joty, G. Carenini, G. Murray, and R. Ng. 2010. Exploiting conversation structure in unsupervised topic segmentation for emails. In *Proc. of EMNLP 2010, Cambridge, MA, USA*.

D. Jurafsky and J. H. Martin, 2008. *Speech and Language Processing*. Prentice Hall.

H. P. Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of Research Development*, 2(2):159–165.

I. Malioutov and R. Barzilay. 2006. Minimum cut model for spoken lecture segmentation. In *Proc. of ACL 2006, Sydney, Australia*.

I. Mani, D. House, G. Klein, L. Hirschman, T. Firmin, and B. Sundheim. 1999. The TIPSTER SUMMAC text summarization evaluation. In *Proc. of EACL 1999, Bergen, Norway*, pages 77–85.

I. Mani. 2001a. *Automatic Summarization*. John Benjamin, Amsterdam, NL.

I. Mani. 2001b. Summarization evaluation: An overview. In *Proc. of the NTCIR Workshop 2 Meeting on Evaluation of Chinese and Japanese Text Retrieval and Text Summarization, Tokyo, Japan*, pages 77–85.

G. Murray, G. Carenini, and R. Ng. 2010. Generating and validating abstracts of meeting conversations: a user study. In *Proc. of INLG 2010, Dublin, Ireland*.

M. Porter, J-P. Onnela, and P. Mucha. 2009. Communities in networks. *Notices of the American Mathematical Society*, 56:1082–1097.

D. Radev, S. Blair-Goldensohn, and Z. Zhang. 2001. Experiments in single and multi-document summarization using MEAD. In *Proc. of DUC 2001, New Orleans, LA, USA*.

W.M. Rand. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:846–850.

E. Shriberg, R. Dhillon, S. Bhagat, J. Ang, , and H. Carvey. 2004. The ICSI meeting recorder dialog act (MRDA) corpus. In *Proceedings of SIGdial Workshop on Discourse and Dialogue, Cambridge, MA, USA*, pages 97–100.

S. Teufel and M. Moens. 1997. Sentence extraction as a classification task. In *Proc. of ACL 1997, Workshop on Intelligent and Scalable Text Summarization, Madrid, Spain*, pages 58–65.