CLASSY 2009: Summarization and Metrics

John M. Conroy Judith D. Schlesinger Dianne P. O'Leary
IDA/Center for Computing Sciences University of Maryland
{conroy, judith}@super.org oleary@cs.umd.edu

1 Introduction

This year the CLASSY team participated in the update summary task and made four submissions to summarization evaluation (AESOP). Our AESOP submissions used combinations of ROUGE scores along with an update (or newness) score. We also use these new metrics, which we call Nouveau ROUGE, to help train our system and evaluate new ideas on computing update summaries.

CLASSY (Clustering, Linguistics, and Statistics for Summarization Yield) is the summarization system developed mainly[1] by IDA/Center for Computing Sciences. Before describing the Nouveau ROUGE update metrics, we describe improvements made to CLASSY in data preparation, redundancy removal, and sentence selection.

2 CLASSY 2009

CLASSY 2009 retains a similar structure of previous versions, consisting of the following seven steps:

1. Data preparation and sentence trimming.
2. Query term selection from the topic descriptions.
3. Signature term computation for each of the document sets.
4. Sentence scoring using the approximate oracle.
5. (For update summaries) Projection of term-sentence matrices against the base summary to reduce redundancy.
6. Redundancy removal and sentence selection.
   (a) Via LSI/L1-QR algorithm followed by an Integer Linear Program (ILP) or

[1] Over the years we've collaborated with several colleagues from the DoD and the University of Maryland.

(b) ILP for redundancy removal and sentence selection.

7. Sentence ordering based on an approximate Traveling Salesperson (TSP) algorithm.

Improvements made to CLASSY this year centered on data preparation, redundancy removal, and sentence selection.

2.1 Data Preparation

We made a significant change to the data preparation step for TAC 2009. Specifically, we made the painful decision to develop a new sentence splitter due to dissatisfaction with those we had tried, including one that was "home-grown". This dissatisfaction was due to speed concerns, for sentence splitters that were POS- or parse-based; and accuracy concerns since mis-split sentences (either run-ons or fragments) affect grammaticality, readability, and even responsiveness. Some splitters exhibited both of these defects to some extent.

Many errors were identified while using previous splitters. Our tokenization routine was adapted to find and correct many of these errors. Unfortunately, this added a complexity to the tokenizer that we would have preferred to avoid. This knowledge, though, was extremely useful while developing our new sentence splitter, F-SPLIT.

F-SPLIT is fast, achieving speeds of between 1000–1100 sentences/second for the entire AQUAINT2 data base. The new sentence splitter is also very accurate with error rates of less than 1% of all identified sentences.

We designed F-SPLIT as a two-pass routine. The nature of F-SPLIT is to split too much, causing sentence fragments, rather than too little, causing run-ons. When the first pass on a long document results in a small number of sentences, a second pass is performed, looking for common causes of errors. This second pass avoids costly, complex evaluation of a file when it most likely is not problematic.

The two-pass approach also turned out to be a boon for making the splitter as accurate as possible. When the initial set of sentences is being read and written to generate the final file, sentences are examined for sentence fragments and run-on sentences. Run-on sentences may be the result of several list items occurring in one sentence or a footnote embedded within a sentence. Sentence fragments are the result of errors such as splits on abbreviations, splits after quotes, spacing errors, and confusion on ellipsis.

A runtime file is generated, identifying every situation where F-SPLIT made a guess while doing the initial split and then each time two sentences were joined or a run-on sentence was split. This file can be used to partially assess the accuracy of the splitter although, of course, there is no way of knowing, other than by painful examination of the input and output, what other errors have been missed.

One example of a correction made by the second pass is to join a quote to the text that indicates the identity of the speaker. In one set of over 423,000 sentences, 282 such joins were made (0.07% of all found sentences). Of those, 279 were correctly joined (98.94%) while 3 were in error (1.06%), for an actual error rate of 0.0007%. For another set of nearly

390,000 sentences, 203 joins were made (0.05% of all sentences) with 198 correctly joined (97.54%) and 5 being in error (2.46%) for an actual error rate of 0.001%.

2.2 Redundancy Removal and Sentence Selection

For the last several years a non-negative (or L1-norm) QR factorization has been at the heart of the CLASSY redundancy removal ([2], [6]). Sentence selection has been done either allowing for truncation or using a simple greedy strategy for finding a subset of sentences that are likely to have a high concentration of terms chosen by humans and that meet the word limit requirement, which is 100 for TAC 2009 [3].

Based on the success of ICSI system [5] at TAC 2008 we decided to experiment with integer linear programming (ILP) methods for redundancy removal and sentence selection. To this end two procedures were evaluated:

- Follow the non-negative QR factorization, which reduces redundancy, with a simple binary linear program to solve a knapsack problem to choose sentences.
- Adapt ICSI's ILP, using an approximate oracle weighting of the terms.

For the first, we use the non-negative QR factorization to select s non-redundant sentences containing at least 120 words. Let n_j denote the number of words in th j-th sentence and $\hat{\omega}_j$ be the approximate oracle score for the j-th sentence, i.e. the term average of the probability that a term will be included in a human-generated summary. We seek to find a subset of the sentences to maximize the total approximate oracle score and contain at most 100 words. If we denote membership in this subset via a binary vector x, then we seek x^* such that

$$x^* = \text{argmax}_x \sum_j x_j \hat{\omega}_j \tag{1}$$

$$\text{subject to} \tag{2}$$

$$\sum_j x_j n_j \leq 100 \tag{3}$$

$$x_j \in \{0, 1\} \tag{4}$$

We solve this problem using Matlab's *bintprog*().

The ICSI ILP addresses the redundancy problem and sentence selection problem simultaneously [5]. We adapt their approach to make use of the approximate oracle score. Experimenting with the TAC 2008 data, we found that passing top-scoring sentences totaling at least 200 words performed the best. We let A be the term-sentence incidence matrix (0's and 1's) of size m by s corresponding to these sentences and n_j the number of words in the j-th sentence. The objective function of the ICSI ILP gives credit for each unique term a sentence contributes to the summary. The inequality constraints ensure that if term

i is included ($y_i = 1$) there is at least one sentence j, that includes it, is included in the summary ($x_i = 1$). Conversely, if a sentence is included in a summary all of its terms are included. We adapt this to the CLASSY scoring by using the estimated probability that a term t for the topic τ will be included in a human summary; this estimate is denoted $\hat{\Pr}(t|\tau)$.

$$y^* = \text{argmax}_y \sum_j y_j \hat{\Pr}(j|\tau) \tag{5}$$

$$\text{subject to} \tag{6}$$

$$\sum_j x_j n_j \leq 100 \tag{7}$$

$$y_i - \sum_j a_{ij} x_j \leq 0 \tag{8}$$

$$a_{ij} x_j - y_i \leq 0 \tag{9}$$

$$y_j \in \{0, 1\} \tag{10}$$

$$x_j \in \{0, 1\} \tag{11}$$

This ILP is larger and more difficult than our first one, but it too can be solved using Matlab's *bintprog*(). In practice, this ILP tends to produce summaries whose average sentence length is shorter and total length is closer to the target length.

3 CLASSY Evaluation Metrics

The CLASSY team developed several methods for evaluating summaries as part of the AESOP task at TAC 2009. For base summaries (subset A), variations of ROUGE were combined via either robust linear regression as studied in [1] or using a linear classifier. For update summaries, we used new ROUGE-based scores which compare the peer summaries for subset B with the model summaries of subset A. We call this newness score "Nouveau ROUGE" [4]. The Nouveau scores can then be combined with ROUGE scores to provide sharper estimates for overall responsiveness or pyramid scores.

3.1 Metrics Based on Robust Linear Regression

Our family of robust linear regression metrics are based upon using average system ROUGE scores to predict either pyramid scoring or overall responsiveness. The features we employed for TAC only include content metrics, so we expect that the induced metrics will only capture the "content responsiveness" of the summaries, since no linguistic information is included. In the future we hope to include linguistic metrics as well.

Formally, the regression problem is given a set of numeric features and the corresponding content responsiveness or pyramid scores. We let a_{ij}, for $i = 1, ..., m$ and $j = 1, .., n$, be the

value of the jth feature for the summarizer i, and let b_i be the human content evaluation metric (e.g., pyramid scoring or overall responsiveness). Canonical correlation finds an $n-$long vector x such that

$$x = \text{argmax } \rho(\sum_{j=1}^{n} a_{ij}x_j, b_i), \tag{12}$$

where ρ denotes the Pearson correlation between two values.

To be robust to outliers we use a robust least squares algorithm (Matlab's $robustfit()$) to minimize $||Ax - b||$, where the norm $||.||$ appropriately weights outliers.

Given a robust linear fit, base on training data, a score for a peer summary to be evaluated is computed by computing the given features for the summary and then applying the linear model x. Thus, resulting score will be estimate the human evaluation metric as function of the computed features.

3.2 Metrics Based on a Linear Classifier

Given class labels (e.g., human responsiveness judgements) for individual summaries and automatically generated features computed for these summaries, a classifier can be built to predict these class labels. We choose to use a classifier for which we can readily compute a posterior probability for the class label. For TAC 2009 we simply used a linear classifier; however, the method generalizes to any classifier for which posterior probabilities can be computed. The posterior probabilities returned by the classifier are then used to compute an expected responsiveness score.

More formally, suppose have a collection of feature matrices A_i where each row corresponds to a summary that was given a categorical human judgement of i and the columns are automatically-computed features. We used TAC 2008 data to train the model, resulting in 5 classes, corresponding to the 5 scores in overall responsiveness.

To train the model we used Matlab's $classify()$ function with the default settings, which assumes a multi-variant normal model for each class and a common covariance matrix. When the model is used to score a summary, it estimates a posterior probability p_i, which is the probability that the summary will have responsiveness score i. The estimated responsiveness score s is simply the expected value, computed as

$$s = \sum_{i=1}^{5} ip_i.$$

4 Results

We now present a synopsis of the CLASSY summarization submission results as well as the AESOP results.

4.1 TAC 2009: Update Summaries

The two CLASSY submissions for TAC 2009 both had the advantage of improved sentence splitting. They differed in their approaches to redundancy removal and sentence selection: submission ID 6 used the non-negative QR factorization for redundancy removal and the simple ILP for sentence selection, while submission ID 54 used an ILP for both redundancy removal and sentence selection.

Unfortunately, it is not possible to do a direct comparison of results from 2008 (or earlier years) to 2009 based on the human linguistic quality scores as the linguistic scoring was changed in 2009 from a 1-5 scale to a 1-10 scale, but reducing the error rates on the training data most likely gave rise to a smaller number of sentence splitting errors.

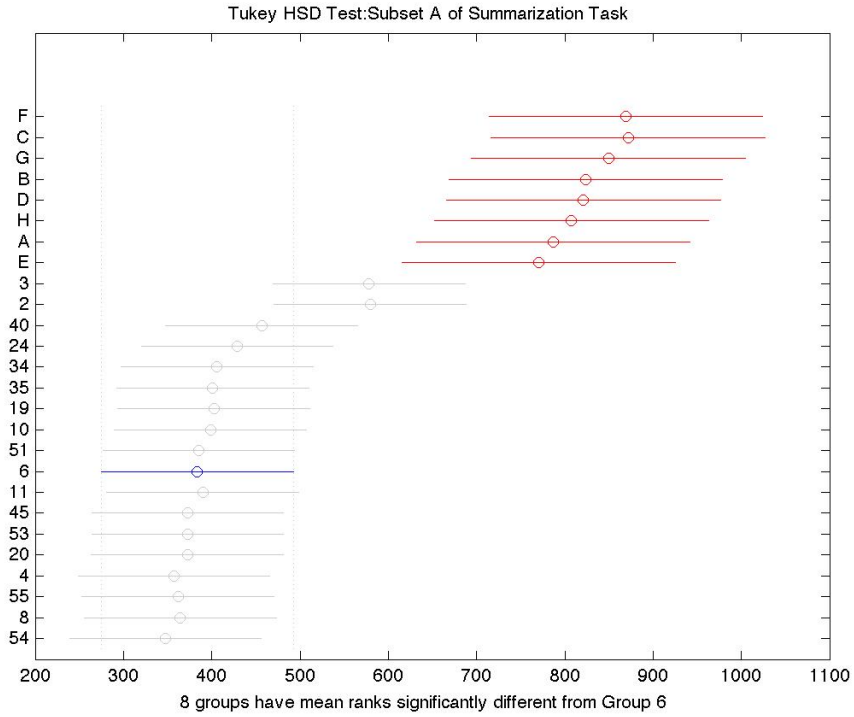
Figure 1 gives the result of a Tukey honestly significant difference test of the top performing systems as evaluated by overall responsiveness. In subset A, we see that only the human summaries score significantly higher than the CLASSY submission. We note that even the human scrambled summaries (baseline 2, the uber-baseline) and the human extracts, baseline 3, overlap the CLASSY interval. For the update task, subset B, the 8 human summarizers outperform all but those two baselines, and baseline 1, the lead of the most recent article, does about as well as all of the submissions.

On linguistic quality, the picture is a bit different. Here we combined both subsets A and B since linguistic quality should not be significantly affected by which task is being performed. We are not surprised to see that the lead, baseline 1, performs well, since it is a human-written summary. We do note that our submission 6 is among the top group of machine systems and is not significantly worse than baseline 1. On the other hand, submission 54 is significantly worse than the baseline. Recall that system 54 tended to choose shorter sentences. We hypothesize that the drop in linguistic quality for this submission is a result of loss of focus, since arranging a larger number of short sentences is more error prone than arranging a smaller number of longer sentences. Furthermore, a sign test comparing the linguistic scores of the two submissions does indeed show that system 6 has significantly higher linguistic scores.

4.2 TAC 2009: AESOP Submissions

For the AESOP task the CLASSY team submitted three runs based on the robust linear regression models and one linear classifier model. All submissions were trained on the TAC 2008 data using the machine-generated summaries only. In each case, the update summaries were scored based on the ROUGE features computed not only for the B subset But also comparing to the A models. Thus, all the models incorporate "nouveau" features. Table 1 gives the labels and characteristics of the submissions.

AESOP had two subtasks. In the first subtask, dubbed "allpeers", systems were required to predict the performance of human systems as well as the machine summaries. Human summaries were compared with the three other models, and for machine summaries



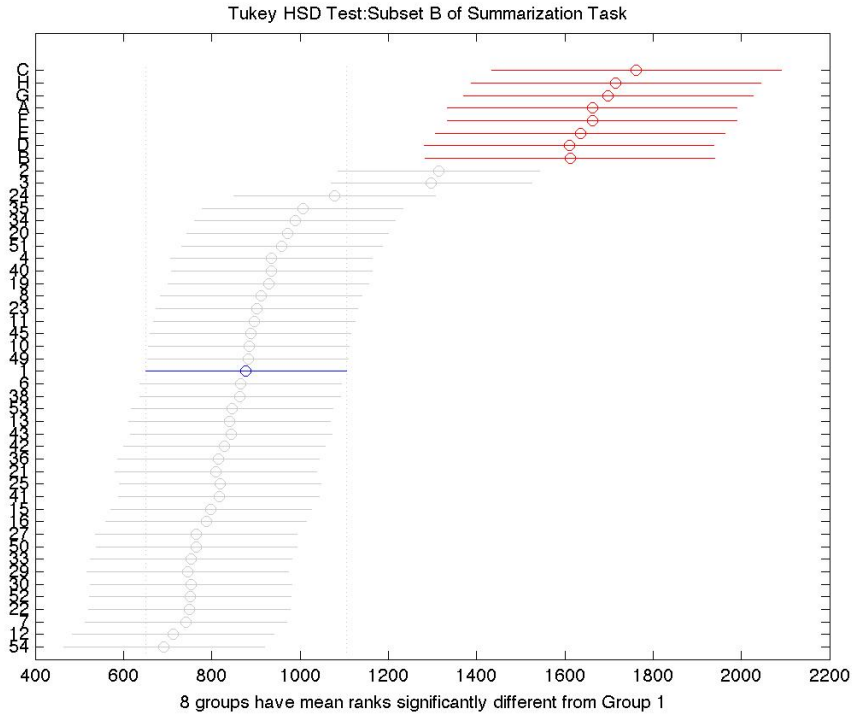


Figure 1: Tukey Honestly Significant Different Test:Overall Responsiveness for Subsets A and B

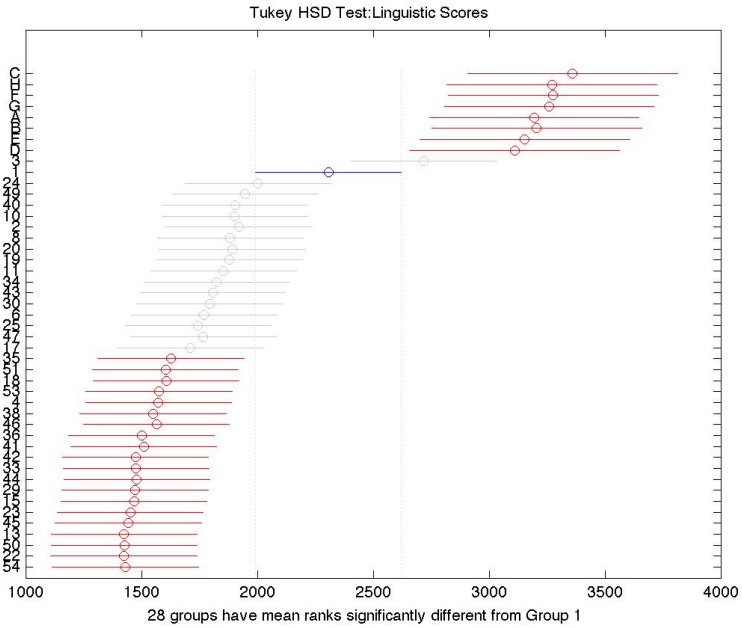


Figure 2: Tukey Honestly Significant Different Test:Linguistic Quality Subsets A and B

Table 1: Four AESOP Submissions by the CLASSY team

System ID	Model Type	ROUGE Features	Target Human Metric
25	Regression	1, 2, 3, L, SU4	Responsiveness
6	Regression	2	Responsiveness
23	Regression	1, 2, 3, L, SU4	Pyramid
26	Classifier	2, 3	Responsiveness

a jackknifing procedure was used to estimate a score. We report Pearson correlation that differs from that computed by NIST; our correlations do not include the uber-baseline or the human summaries. The former is excluded because the scores for it are skewed since it is always compared with a scrambled version of itself. Human summaries were excluded since the CLASSY metrics were trained simply on machine summaries and are intended to distinguish among this group. To adequately score human summaries, linguistic measures would need to be added to the feature sets.

Tables 2 and 3 give the Pearson correlation for the top performing metrics for the subsets A and B. We note submission 6, which for subset A is simply ROUGE-2, gives the best prediction for the pyramid scores. System 26, the linear classifier trained on responsiveness, performs just slightly worse on this sub-task. Conversely, the ROSE submissions 25 and 15 are the top two predictors for overall responsiveness. For the update summaries, submission 15 is the best predictor for overall responsiveness.

The second sub-task for AESOP focused simply on prediction of machine system performance. This task is labeled "nomodels." Here all 4 human models can be used at once to compute the metrics. Tables 4 and 5 give the Pearson correlation results for the top performing metrics. The CLASSY submissions f=or this sub-task performed comparably, with the same ranking as with the "allpeers" sub-task. (We illustrate the importance of removing the uber-baseline from the results and note that if it were not removed system 15 drops from best predicator in the subset B to 23rd best for pyramid and 28th best for responsiveness! Systems 11 and our "beloved" system 26 are the best predictors for pyramid and responsiveness. One outlier greatly changes the results of Table 5. The update task data appears to be more sensitive to this outlier as the best)

5 Conclusion

The two CLASSY summarization entries performed within the top group of machine-generated summaries. However, while even the top machine-generated summaries score near human performance in the classic ROUGE metrics, they are far from human performance in overall responsiveness. Sadly, no system beat the baseline for the update task.

The three of the four our submissions for AESOP were the best predictors for base summaries. These new metrics as well as more research in linguistics features should help developers continue to make advances in automatic text summarization.

References

[1] John M. Conroy and Hoa Trang Dang. Mind the Gap: Dangers of Divorcing Evaluations of Summary Content from Linguistic Quality. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 145–152, Manchester, UK, August 2008.

Table 2: All Peers subset A

Jackknifed Pyramid				Overall Responsiveness			
ID	ρ	LB	UB	ID	ρ	LB	UB
6	0.972	0.952	0.984	25	0.890	0.817	0.935
26	0.968	0.945	0.981	15	0.880	0.801	0.929
15	0.967	0.943	0.981	28	0.876	0.794	0.926
28	0.966	0.942	0.980	12	0.875	0.793	0.926
10	0.959	0.931	0.976	23	0.875	0.793	0.926
11	0.957	0.927	0.975	11	0.872	0.788	0.924
13	0.954	0.921	0.973	6	0.872	0.788	0.924
12	0.954	0.921	0.973	26	0.871	0.786	0.923
4	0.951	0.917	0.972	10	0.863	0.774	0.918
1	0.949	0.914	0.970	31	0.863	0.774	0.918
2	0.948	0.912	0.970	5	0.860	0.770	0.917
19	0.948	0.911	0.969	27	0.857	0.765	0.915
25	0.947	0.910	0.969	33	0.855	0.761	0.913
24	0.947	0.910	0.969	16	0.853	0.758	0.912
16	0.946	0.908	0.968	17	0.852	0.757	0.912
18	0.946	0.908	0.968	19	0.852	0.757	0.912
5	0.945	0.906	0.968	4	0.851	0.755	0.911
33	0.944	0.905	0.967	2	0.849	0.753	0.910
32	0.937	0.894	0.963	13	0.849	0.752	0.910
17	0.935	0.890	0.962	18	0.848	0.750	0.909
21	0.929	0.880	0.958	8	0.841	0.739	0.905
31	0.927	0.877	0.957	1	0.839	0.737	0.904
27	0.927	0.876	0.957	24	0.823	0.712	0.894
23	0.926	0.875	0.956	34	0.806	0.686	0.883

Table 3: All Peers subset B (Update)

Jackknifed Pyramid				Overall Responsiveness			
ID	ρ	LB	UB	ID	ρ	LB	UB
15	0.969	0.947	0.982	15	0.871	0.787	0.923
28	0.969	0.946	0.982	28	0.859	0.767	0.916
10	0.961	0.933	0.977	10	0.855	0.761	0.913
2	0.956	0.925	0.974	23	0.852	0.757	0.912
12	0.951	0.917	0.972	31	0.847	0.749	0.909
11	0.943	0.904	0.967	2	0.846	0.748	0.908
25	0.943	0.903	0.967	12	0.842	0.741	0.905
26	0.941	0.901	0.966	11	0.828	0.720	0.897
6	0.940	0.899	0.965	33	0.826	0.718	0.896
24	0.939	0.897	0.965	26	0.820	0.707	0.892
16	0.936	0.892	0.963	6	0.816	0.702	0.890
33	0.936	0.892	0.963	5	0.813	0.696	0.887
19	0.935	0.891	0.962	25	0.808	0.690	0.885
23	0.932	0.886	0.960	19	0.803	0.683	0.882

Table 4: No Models subset A

Jackknifed Pyramid				Overall Responsiveness			
ID	ρ	LB	UB	ID	ρ	LB	UB
6	0.972	0.952	0.984	25	0.888	0.814	0.934
26	0.967	0.944	0.981	15	0.880	0.801	0.929
15	0.967	0.944	0.981	28	0.876	0.794	0.926
28	0.967	0.943	0.981	12	0.875	0.793	0.926
10	0.960	0.931	0.977	23	0.875	0.793	0.926
25	0.959	0.930	0.976	11	0.872	0.788	0.924
11	0.958	0.929	0.976	6	0.872	0.788	0.924
13	0.958	0.928	0.975	26	0.871	0.787	0.924
12	0.954	0.921	0.973	10	0.863	0.774	0.918
4	0.952	0.918	0.972	31	0.863	0.774	0.918
19	0.950	0.915	0.971	5	0.860	0.770	0.917
1	0.950	0.914	0.971	27	0.857	0.765	0.915
2	0.948	0.912	0.970	33	0.855	0.761	0.913
24	0.947	0.911	0.969	16	0.853	0.758	0.912
18	0.947	0.910	0.969	17	0.852	0.757	0.912
16	0.947	0.910	0.969	4	0.851	0.755	0.911
5	0.945	0.907	0.968	19	0.850	0.753	0.910
33	0.945	0.906	0.968	2	0.849	0.752	0.910
23	0.943	0.904	0.967	18	0.844	0.744	0.907

Table 5: No Models subset B (Update)

Jackknifed Pyramid				Overall Responsiveness			
ID	ρ	LB	UB	ID	ρ	LB	UB
15	0.969	0.947	0.982	15	0.871	0.787	0.923
28	0.969	0.946	0.982	28	0.859	0.767	0.916
10	0.961	0.934	0.977	10	0.855	0.761	0.913
2	0.956	0.925	0.974	25	0.847	0.749	0.909
12	0.951	0.917	0.972	31	0.847	0.749	0.909
25	0.947	0.910	0.969	2	0.846	0.748	0.908
11	0.944	0.905	0.967	23	0.845	0.746	0.908
6	0.940	0.899	0.965	12	0.842	0.741	0.905
24	0.940	0.899	0.965	11	0.828	0.720	0.897
26	0.938	0.895	0.964	33	0.826	0.718	0.896
16	0.937	0.893	0.963	26	0.818	0.705	0.891
33	0.937	0.893	0.963	6	0.817	0.703	0.890
19	0.934	0.888	0.961	5	0.813	0.696	0.887
23	0.933	0.887	0.961	19	0.804	0.683	0.882

[2] John M. Conroy, Dianne P. O'Leary, and Judith D. Schlesinger. CLASSY Arabic and English multi-document summarization. In *Multi-Lingual Summarization Evaluation 2006*, 2006. `http://www.isi.edu/~cyl/MTSE2006/MSE2006/papers/index.html`.

[3] John M. Conroy and Judith D. Schlesinger. CLASSY and TAC 2008 metrics, 2008.

[4] John M. Conroy, Judith D. Schlesinger, and Dianne P. O'Leary. Nouveau rouge: A novelty metric for update summarization. 2009.

[5] Dan Gillick, Benoit Favre, and Dilek Hakkani-Tür. The ICSI summarization system at TAC 2008, 2008.

[6] Judith D. Schlesinger, Dianne P. O'Leary, and John M. Conroy. Arabic/english multi-document summarization with classy - the past and the future. In Alexander F. Gelbukh, editor, *CICLing*, volume 4919 of *Lecture Notes in Computer Science*, pages 568–581. Springer, 2008.