AutoSummENG and MeMoG in Evaluating Guided Summaries

George Giannakopoulos
NCSR Demokritos, Greece
ggianna@iit.demokritos.gr

Vangelis Karkaletsis
NCSR Demokritos, Greece
vangelis@iit.demokritos.gr

Abstract

Within this article, we present the application of the AutoSummENG and MeMoG methods within the TAC 2011 AESOP challenge. Both evaluation methods are based on n-gram graphs. The experiments indicate that both methods offer very high performance in different aspects of evaluation, without the need of deep analysis or preprocessing. The results also imply some interesting open problems and point to further directions of study, related to negative examples of good summaries.

1 Introduction

Automatic methods for the evaluation of summaries exist for some time now (Hovy et al., 2005b; Lin, 2004; Zhou et al., 2006; Schilder and Kondadadi, 2009) and correlate highly to the measure of *responsiveness*. This measure, first appearing within the DUC community (see *e.g.,* (Dang, 2005)), is expected to represent a measure of information completeness and linguistic quality for a given text, as a human assesses it. Even though the correlation of the automatic methods to the human grades is high on the system level, until recently there were some other desired characteristics that did not coexist in a single automatic method. More precisely:

- No preprocessing. A method that does not require language-dependent preprocessing or resources (thesauri, lexica, etc.).
- Full automation. A method should not require human intervention, apart from the human model summaries.
- Context-sensitivity. A method should take into account contextual information, so that well-formedness of text is taken into account. Well-formedness can be loosely defined as the quality of a text that allows easy reading. A text that is a random sequence of words would lack this quality, even if the words are on topic.

The AutoSummENG method (Giannakopoulos et al., 2008) (AUTOmatic SUMMary Evaluation based on N-gram Graphs), holds all these qualities, while bearing results with high correlation to the responsiveness measure, which indicates correlation to human judgment.

In the following paragraphs, we introduce the basic notions and algorithms for the representation of texts through n-gram graphs (Section 2). We then describe the methodology for the comparison and how this comparison leads to the grading of summaries and systems (Section 4). We present experimental results on the TAC2011 AESOP task (Section 5) and conclude the paper (Section 6) with the lessons learned and future directions.

2 System Overview

The AutoSummENG system (Giannakopoulos et al., 2008) is based upon the JInsect library[1] of "n-gram graph"-based text processing. For our experiments in TAC 2011, we applied the AutoSummENG method, as well as the Merge Model Graph (MeMoG) variation over the TAC 2011 AESOP task data. Thus, the study refers to two methods:

[1] See http://sourceforge.net/projects/jinsect and http://www.ontosum.org for more information.

- The first method is the original AutoSummENG, for default parameters of L_{min}, L_{MAX} and D_{win} for this year's corpus. This method creates an n-gram graph representation of the evaluated text and another n-gram graph per model summary. Then, the measure of Value Similarity is used to compare the similarity of the evaluated text to each model summary. The average of these similarities is considered to represent the overall performance of the summary text.

- The second method, instead of comparing the graph representation of the evaluated summary text to the graph representation of individual model texts and averaging over them, calculates the *merged* graph of all model texts. Then, it compares the evaluated summary graph to this overall model graph. We term this variation the Merged Model Graph method (MeMoG) and it aims to non-linearly combine the content of texts into one representative whole.

In order to introduce the reader to the method and its alternatives, we need to recapitulate the basic concepts of AutoSummENG and the n-gram graph representation theory.

3 Representation and Basic Algorithms

In the domain of natural language processing, there have been a number of methods using *n-grams*. An n-gram is a, possibly ordered, set of words or characters, containing n elements (see Example 3.1). N-grams have been used in summarization and summary evaluation (Banko and Vanderwende, 2004; Lin and Hovy, 2003; Copeck and Szpakowicz, 2004). In the automatic summarization domain, n-grams mostly appear as *word* n-grams, as happens in the ROUGE/BE family of evaluator methods (Hovy et al., 2005a; Lin, 2004).

Example 3.1 Examples of n-grams from the sentence: *This is a sentence.*
Word unigrams: this, is, a, sentence
Word bi-grams: this is, is a, a sentence
Character bi-grams: th, hi, is, s␣, ␣a, ...
Character 4-grams: this, his␣, ␣is␣, ...

3.1 Extracting N-grams

To extract the n-grams (S^n) of a text T^l, we follow the (elementary) algorithm indicated as algorithm 1. The algorithm's complexity is linear to the size $|T|$ of the input text T.

Input: text T
Output: n-gram set SS^n
`// T is the text we analyze`
1 $SS^n \leftarrow \emptyset$;
2 **for** *all i in [1,length(T)-n+1]* **do**
3 $\quad SS^n \leftarrow SS^n \cup S_{i,i+n-1}$
4 **end**

Algorithm 1: Extraction of n-grams

The algorithm applies no preprocessing (such as extraction of spaces, punctuation or lemmatization). Furthermore, it obviously extracts overlapping parts of text, as the sliding window of size n is shifted by one position and not by n positions at a time. This technique is used to avoid the problem of segmenting the text. The redundancy apparent in this approach proves to be useful *similarly to a convolution function*: summing similarities over a scrolling window may prove useful when you do not know exactly where to start matching two strings.

In the case of summary evaluation we may compare common n-grams between a peer (judged) summary and a model summary. The extracted, overlapping n-grams are certain to match corresponding n-grams of the model summary, if such n-grams exist. That would not be the case for a method where the text would be segmented in equally sized n-grams.

Example 3.2 *Application of our method to the sentence we have used above, with a requested n-gram size of 3 would return:*
{*'Do ', 'o y', ' yo', 'you', 'ou ', 'u l', ' li', 'lik', 'ike', 'ke ', 'e t', ' th', 'thi', 'his', 'is ', 's s', ' su', 'sum', 'umm', 'mma', 'mar', 'ary', 'ry?'*}
while an algorithm taking disjoint n-grams would return
{*'Do ', 'you', ' li', 'ke ', 'thi', 's s', 'umm', 'ary'*}
(and '?' would probably be omitted). The segmentation has reduced the number of existing n-grams examined, based on the disjointness prerequisite.

The *n-gram graph* is a graph $G = \{V^G, E^G, L, W\}$, where V^G is the set of ver-

tices, E^G is the set of edges, L is a function assigning a label to each vertex *and to each edge* and W is a function assigning a weight to every edge. The graph has n-grams as its vertices $v^G \in V^G$. The edges $e^G \in E^G$ (the superscript G will be omitted where easily assumed) connecting the n-grams indicate proximity of the corresponding vertex n-grams.

The edges are weighted by the number of co-occurrences of their respective n-grams within a given window in the source text of the graph. We note that the meaning of distance and window size changes by whether we use character or word n-grams. In this work we use character n-grams, which have been shown to perform better than their word counterpart (Giannakopoulos et al., 2008).

The labeling function L for edges assigns to each edge the concatenation of the labels of its corresponding vertices' labels in a predefined order: for directed graphs the order is the order of the edge direction while in undirected graphs the order can be the lexicographic order of the vertices' labels. To ensure that no duplicate vertices exist, we require that the labeling function is an one-to-one function.

More formally:

Definition 3.3 *if* $S = \{S_1, S_2, ...\}, S_k \neq S_l$, *for* $k \neq l, k, l \in \mathbb{N}$ *is the set of* distinct *n-grams extracted from a text* T^l, *and* S_i *is the* i*-th extracted n-gram, then* $G = \{V^G, E^G, L, W\}$ *is a graph where* $V^G = S$ *is the set of vertices* v, E^G *is the set of edges* e *of the form* $e = \{v_1, v_2\}$, $L : V^G \rightarrow \mathbb{L}$ *is a function assigning a label* $l(v)$ *from the set of possible labels* $\mathbb{L}$ *to each vertex* v *and* $W : E^G \rightarrow \mathbb{R}$ *is a function assigning a weight* $w(e)$ *to every edge.*

In our implementation, the edges E are assigned weights of $c_{i,j}$ where $c_{i,j}$ is the number of times a given pair S_i, S_j of n-grams happen to be neighbors in a string within some distance D_{win} of each other. The distance between two strings in a text is the absolute difference of the positions of their first characters in the text. The vertices v_i, v_j corresponding to n-grams S_i, S_j that are located within this distance D_{win} are connected by a corresponding edge $e \equiv \{v_i, v_j\}$.

The n-gram graph extracted with the above process from a given string, actually represents a compressed form of the string. In fact, the graph is a model of the string, describing the neighborhood relations between n-grams in the string. Each edge describes one restriction, based on the n-gram graph parameters n, D_{win}. Thus, e.g., the edge $e = < a, b, 2.0 >$ in a graph G with $n = 1, D_{\text{win}} = 3$ declares that the text the edge came from contains two co-occurrences of letters a and b within a distance of 3 from each other.

From a given graph G, in the general case, we can generate a multitude of texts. These texts are the set of possible texts that may conform to all the neighborhood restrictions that the graph edges imply. As an example:

Example 3.4 *The trivial, symmetric, n-gram graph of* $n = 1$, $D_{win} = 1$, *with* $V = \{a, b\}, E = \{< a, b, 1.0 >\}$ *has two possible source texts: ab, ba.*

This lack of one-to-one mapping between graph and text offers some generalization ability to the n-gram graph representation, which can be important when comparing texts that may have been expressed similarly, but not identically.

The set of texts an n-gram graph represents cannot be trivially calculated: it is a Constraint Satisfaction problem, with each edge posing a constraint. It is also possible that a given n-gram graph can represent no texts, if there exist no solutions that conform to all the edge-implied constraints. Of course, an n-gram graph that is created from a given text, always represents at least one text: its source.

In the summary evaluation domain, we can represent peer summaries and model summaries as n-gram graphs and compare them in the n-gram graph domain, taking advantage of the n-gram graph generalization ability. However, if one has more than one model summary available, one may wish to create a representative graph for *all model summaries together*. This can be achieved as described in the following section, through the Merged Model Graph.

3.2 MeMoG: The Merged Model Graph

Given two instances of n-gram graph representation G_1, G_2, there is a number of operators that can be applied on G_1, G_2 to provide the n-gram graph equivalent of union, intersection and other such operators of set theory (Giannakopoulos, 2009). In

our applications we have used the *update operator* U (similar to the merging operator), which allows the creation of a "centroid" graph. The update function $U(G_1, G_2, l)$ takes as input two graphs, one that is considered to be the pre-existing graph G_1 and one that is considered to be the new graph G_2. The function also has a parameter called the *learning factor* $l \in [0, 1]$, which determines the sensitivity of G_1 to the change G_2 brings.

Focusing on the weighting function of the graph, resulting from the application of $U(G_1, G_2, l)$, the higher the value of learning factor, the higher the impact of the new graph to the resulting graph of the update. More precisely, a value of $l = 0$ indicates that G_1 will completely ignore the (considered new) graph G_2. A value of $l = 1$ indicates that the weights of the edges of G_1 will be assigned the values of the new graph's edges' weights. A value of 0.5 gives us the merging operator. The definition of the weighting performed in the graph resulting from U is:

$$W^i(e) = W^1(e) + (W^2(e) - W^1(e)) \times l \quad (1)$$

The U function allows using graphs to model a whole *set* of documents: in our case the model set. The model graph creation process comprises the initialization of a graph with the first document of the model set and the updating of that initial graph with the graphs of following model summaries. Especially, when one wants the overall graph's edges to hold weights averaging the weights of all the individual graphs that have contributed to it, then the i-th new graph that updates the overall graph should use a learning factor of $l = \frac{1}{i}, i > 1$. This gives a graph that has a role similar to the centroid of a set of vectors: it functions as a representative graph for the set its constituent graphs.

4 From Graph Matching to Summary Evaluation

Graph similarity calculation methods can be classified into two main categories.

Isomorphism-based Isomorphism is a bijective mapping between the vertex set of two graphs V_1, V_2, such that all mapped vertices are equivalent, and every pair of vertices from V_1 shares the same state of neighborhood, as their corresponding vertices of V_2. In other words, in two isomorphic graphs all the nodes of one graph have their unique equivalent in the other graph, and the graphs also have identical connections between equivalent nodes. Based on the isomorphism, a *common subgraph* can be defined between V_1, V_2, as a subgraph of V_1 having an isomorphic equivalent graph V_3, which is a subgraph of V_2 as well. The *maximum common subgraph* of V_1 and V_2 is defined as the common subgraph with the maximum number of vertices. For more formal definitions and an excellent introduction to the error-tolerant graph matching, *i.e.,* fuzzy graph matching, see (Bunke, 1998).

Given the definition of the maximum common subgraph, a series of distance measures have been defined using various methods of calculation for the maximum common subgraph, or similar constructs like the Maximum Common Edge Subgraph, or Maximum Common Induced Graph (also see (Raymond et al., 2002)).

Edit-distance Based Edit distance has been used in fuzzy string matching for some time now, using many variations (see (Navarro, 2001) for a survey on approximate string matching). The edit distance between two strings corresponds to the minimum number of edit character operations (namely insertion, deletion and replacement) needed to transform one string to the other. Based on this concept, a similar distance can be used for graphs (Bunke, 1998). The edit operations for graphs' nodes are *node* deletion, insertion and substitution. The same three operations can by applied on edges, giving *edge* deletion, insertion and substitution.

Using a transformation from text to graph, the aforementioned graph matching methods can be used as a means to indicate text similarity.

We have applied the Value Similarity calculation (Giannakopoulos et al., 2008), which offers graded similarity indication between two document graphs. Moreover, in order to compare a whole set of documents (model summaries) to a single evaluated text (evaluated summary) we represent the set of docu-

ments with a single graph, as we show in the following sections, whether be it an n-gram graph or a hierarchical proximity graph.

To compare two texts (or character sequences in general) T_1 and T_2 *e.g.,* for the task of summary evaluation against a gold standard text, we need to compare the texts' representations. Given that the representation of a text T_i is a set of graphs $\mathbb{G}_i$, containing graphs of various ranks, we use the *Value Similarity (VS)* for *every n-gram rank*, indicating how many of the edges contained in graph G^i are contained in graph G^j, considering also the weights of the matching edges. In this measure each matching edge e having weight w_e^i in graph G^i contributes $\frac{\text{VR}(e)}{\max(|G^i|,|G^j|)}$ to the sum, while not matching edges do not contribute (consider that for an edge $e \notin G^i$ we define $w_e^i = 0$). The *ValueRatio (VR)* scaling factor is defined as:

$$\text{VR}(e) = \frac{\min(w_e^i, w_e^j)}{\max(w_e^i, w_e^j)} \tag{2}$$

The equation indicates that the *ValueRatio* takes values in $[0, 1]$, and is symmetric. Thus, the full equation for *VS* is:

$$\text{VS}(G^i, G^j) = \frac{\sum_{e \in G^i} \frac{\min(w_e^i, w_e^j)}{\max(w_e^i, w_e^j)}}{\max(|G^i|, |G^j|)} \tag{3}$$

VS is a measure converging to 1 for graphs that share both the edges and similar weights, which means that a value of $\text{VS} = 1$ indicates perfect match between the compared graphs. Another important measure is the *Normalized Value Similarity (NVS)*, which is computed as:

$$\text{NVS}(G^i, G^j) = \frac{VS}{\frac{\min(|G^i|,|G^j|)}{\max(|G^i|,|G^j|)}} \tag{4}$$

The fraction $\text{SS}(G^i, G^j) = \frac{\min(|G^i|,|G^j|)}{\max(|G^i|,|G^j|)}$, is also called Size Similarity. The NVS is a measure of similarity where the ratio of sizes of the two compared graphs does not play a role. In the TAC case there is no real difference, however, because the SS factor is almost constant and equal to 1: the summaries have an almost fixed size. Thus, VS is equivalent to NVS.

The overall similarity VS^O of the sets $\mathbb{G}_1, \mathbb{G}_2$ is computed as the weighted sum of the VS over all ranks:

$$\text{VS}^O(\mathbb{G}_1, \mathbb{G}_2) = \frac{\sum_{r \in [L_{\min}, L_{\text{MAX}}]} r \times \text{VS}^r}{\sum_{r \in [L_{\min}, L_{\text{MAX}}]} r} \tag{5}$$

where VS^r is the VS measure for extracted graphs of rank r in $\mathbb{G}$, and $L_{\min}$, L_{MAX} are arbitrary chosen minimum and maximum n-gram ranks.

The similarity function calculation has a complexity of $O(|G_1| \times |G_2|)$, due to the fact that for each edge in G_1 one needs to lookup its identical edge in G_2. The similarity function calculation has a complexity of $O(|G_1| \times |G_2|)$, due to the fact that for each edge in G_1 one needs to lookup its identical edge in G_2. If an index is maintained with the edges' labels or the vertices' labels, this complexity can be diminished, which is the case in our implementation. Therefore, for every edge in the smallest of the two graphs, we perform a low complexity lookup in the edges of the biggest graph. If an edge is found we perform the calculation of the edge's contribution to the similarity sum. Otherwise, we continue with the next edge from the small graph. This gives a real complexity that is $O(h\min(|G_1|, |G_2|))$, where h is the constant time for a hash map lookup, if the edges are hashed. If the vertices are hashed, then the complexity is $O(h \min(|G_1|, |G_2|) \operatorname{degree}(G_2))$, where the $\operatorname{degree}(G_2)$ function returns the maximum number of edges connected to a single node in G_2.

In the AutoSummENG case, the grade of a summary is the average of the similarities to the model summaries — using the corresponding n-gram graph representations. In the case of MeMoG, there is only one similarity measurement, which is set to be the summary score. The performance of a summarization system is calculated as the average of its summary scores.

5 Experiments

The presented methods have been applied as part of the Automatically Evaluating Summaries Of Peers (AESOP) task of TAC2011:

> The AESOP task is to automatically score a summary for a given metric. AESOP complements the basic summarization task by building a collection of automatic evaluation tools that support development of summarization systems.

More precisely, the purpose of the AESOP task was "to create an automatic scoring metric for summaries, that would correlate highly with one or more of three manual methods of evaluating summaries, as applied in the TAC 2012 Guided Summarization task". The manual methods were the following:

- Pyramid method (modified pyramid score) (Passonneau et al., 2006).
- Overall Readability: NIST described this measure, based on the assessors guidelines, as follows: "The assessor will give a readability/fluency score to each summary. The score reflects the fluency and readability of the summary (independently of whether it contains any relevant information) and is based on factors such as the summary's grammaticality, non-redundancy, referential clarity, focus, and structure and coherence"[2].
- Overall Responsiveness (Dang and Owczarzak, 2008).

The scoring metrics (better "measures") are to evaluate summaries including both model (*i.e.,* human generated) and automatic (non-model) summaries, produced within the TAC 2011 Guided Summarization task[3].

In the Guided Summarization task, 8 human summarizers produced a total of 352 model summaries, and 51 automatic summarizers produced a total of 4,488 automatic summaries. Included in the set of automatic summarizers were two baseline summarizers. The first returns all the leading sentences (up to 100 words) in the most recent document. Summarizer 2 is actually the MEAD automatic summarizer[4], with all default settings. The set of automatic summarizers for AESOP also contains a completely unresponsive "dummy" summarizer from NIST, under Summarizer ID 7.

The summaries are split into Main (or Initial) Summaries (Set A) and Update Summaries (Set B), according to the part of the Guided Summarization Task they fall into[5].

The experiments conducted upon the TAC 2011 corpus were based on the application of the AutoSummENG and the MeMoG methods on the TAC corpus. The n-gram graph parameters used for AutoSummENG were $(L_{\text{min}}, L_{\text{MAX}}, D_{\text{win}}) = (3, 3, 3)$, which seems to offer near-optimal results on many English corpora. The same parameters were used for the MeMoG case.

We note that there were two different types of evaluation: the *All Peers* and the *No Models* evaluation. In the *No Models* case, the peer summaries are evaluated against all model summaries. In the *All Peers* case, the model summaries are evaluated against the remaining model summaries; on the other hand, the peer summaries are evaluated using jack-knifing against all the model summaries. The process of jack-knifing is the following. Given that there are 4 model summaries to evaluate against, the peer summary is evaluated against all combinations of the 4 models in groups of 3. The average grade assigned by all the evaluations is assigned to be the grade of the summary. The results of the evaluation, concerning the correlation to the Pyramid score are shown in Tables 1, 2; the results concerning the correlation to the Overall Responsiveness score are shown in Tables 3, 4; the results concerning the correlation to the Readability score are shown in Tables 5, 6. All the correlation tests gave a p-value $< 10^{-2}$, making the results statistically significant.

Overall, the performance of both AutoSummENG and MeMoG, when related to ranking (Kendall's tau) of All Peers is within the first 4 ranks between the 25 systems in the AESOP task for Group A over all peers. However, the performance for Group B is significantly lower and provides some interesting ground for research: what is there in Group B that lowers the performance that much.

Furthermore, the performance drop is increased in the No Models case, where we see the (baseline) ROUGE offer very high performance. What this may imply is that we need to improve our criteria when judging performances between automatic systems, i.e., when judging the *importance of dif-*

[2] From http://www.nist.gov/tac/2011/Summarization/Guided-Summ.2011.guidelines.html on Oct, 20th 2011.

[3] See http://www.nist.gov/tac/2011/Summarization/Guided-Summ.2011.guidelines.html for more details on the Guided Summarization task.

[4] MEAD summarizer v 3.12, publically available at http://www.summarization.com/mead/.

[5] See http://www.nist.gov/tac (Last visit: Feb 14, 2011) for more info on the Guided Summarization Task of TAC 2011.

Variant (ID)	Pearson (Rank)	Spearman (Rank)	Kendall (Rank)
Group A - All Peers			
AutoSummENG (12)	0.842 (10)	0.932 (3)	0.798 (2)
MeMoG (18)	0.951 (4)	0.924 (6)	0.776 (6)
Group B - All Peers			
AutoSummENG (12)	0.824 (10)	0.885 (3)	0.723 (2)
MeMoG (18)	0.933 (7)	0.888 (2)	0.723 (2)

Table 1: Correlation of grades to Pyramid score for All Peers

Variant (ID)	Pearson (Rank)	Spearman (Rank)	Kendall (Rank)
Group A - No Models			
AutoSummENG (12)	0.974 (2)	0.897 (6)	0.747 (4)
MeMoG (18)	0.964 (8)	0.888 (10)	0.723 (11)
Group B - No Models			
AutoSummENG (12)	0.856 (11)	0.825 (5)	0.640 (5)
MeMoG (18)	0.885 (8)	0.824 (6)	0.640 (5)

Table 2: Correlation of grades to Pyramid score without models

Variant (ID)	Pearson (Rank)	Spearman (Rank)	Kendall (Rank)
Group A - All Peers			
AutoSummENG (12)	0.815 (10)	0.899 (1)	0.748 (1)
MeMoG (18)	0.963 (3)	0.893 (3)	0.727 (3)
Group B - All Peers			
AutoSummENG (12)	0.774 (10)	0.908 (3)	0.753 (3)
MeMoG (18)	0.975 (1)	0.889 (5)	0.726 (6)

Table 3: Correlation of grades to Overall Responsiveness score for All Peers

Variant (ID)	Pearson (Rank)	Spearman (Rank)	Kendall (Rank)
Group A - No Models			
AutoSummENG (12)	0.947 (7)	0.845 (1)	0.675 (1)
MeMoG (18)	0.948 (6)	0.845 (2)	0.671 (2)
Group B - No Models			
AutoSummENG (12)	0.873 (11)	0.861 (5)	0.686 (5)
MeMoG (18)	0.891 (8)	0.838 (10)	0.668 (9)

Table 4: Correlation of grades to Overall Responsiveness score without models

Variant (ID)	Pearson (Rank)	Spearman (Rank)	Kendall (Rank)
Group A - All Peers			
AutoSummENG (12)	0.770 (10)	0.674 (1)	0.513 (2)
MeMoG (18)	0.906 (3)	0.651 (6)	0.494 (5)
Group B - All Peers			
AutoSummENG (12)	0.761 (10)	0.609 (6)	0.465 (5)
MeMoG (18)	0.895 (5)	0.595 (7)	0.432 (7)

Table 5: Correlation of grades to Readability score for All Peers

Variant (ID)	Pearson (Rank)	Spearman (Rank)	Kendall (Rank)
Group A - No Models			
AutoSummENG (12)	0.794 (2)	0.497 (1)	0.359 (2)
MeMoG (18)	0.791 (3)	0.475 (3)	0.351 (3)
Group B - No Models			
AutoSummENG (12)	0.644 (8)	0.399 (4)	0.306 (3)
MeMoG (18)	0.670 (5)	0.394 (5)	0.286 (5)

Table 6: Correlation of grades to Readability score without models

ferent problems in summaries between non-perfect summarizers.

This may well be related to grading inappropriate content, which may need anti-model summaries, providing (possibly) varying cases of bad summaries. It is possible that using varieties of bad summaries (one without focus, one without coherence, one with bad content, one with mispellings and so forth) may provide the additional axes that we can use to define quality (and the lack of it).

In Tables 7, 8 we illustrate the number of agreements and disagreements (see Appendix A for their definition) between measures, on whether the difference in performance between systems is statistically significant or not.

Thus, the first columns of the first row in Table 7 indicates that the verdict that came by using AutoSummENG grades to determine statistically significant difference in performance between systems, agrees 195 times with the verdict when using Pyramid score, while it disagrees 213. There are also columns for the Responsiveness and the Readability measure and there is differentiation between Group A (Main Summaries) and Group B (Update Summaries) texts.

In the All Peers case, MeMoG is the definite winner between AutoSummENG and MeMoG with only a single disagreement concerning the Readability measure. If not for this one error, MeMoG would have performed flawlessly.

In the NoModels case, things are not as clear: in Group A, AutoSummENG performs better, while MeMoG wins the day in Group B. Both methods perform well — 1 disagreement for every 5 agreements — concerning the Pyramid and responsiveness measures. The ratio falls to 1 disagreement for every 3 agreements in the Readability case, which however still remains an acceptable ratio. We stress that across all measures and all settings (No Model vs. All Peers) there were only 3 cases where either of the measures indicated statistically significant difference between two systems, but inversed ranking. This means that most disagreements were on whether a system was significantly better than another and not about which one of the two was better.

6 Conclusion

This paper briefly presented the results of applying AutoSummENG and MeMoG methods for summarization system evaluation on the TAC 2011 corpus. Both methods offered very good results in different aspects of the evaluation.

In the future, we plan to combine all the n-gram graph based methods (AutoSummENG, MeMoG and HPG (Giannakopoulos and Karkaletsis, 2010)), through machine learning to devise an overall evaluation methodology, adopting to individual target measures (e.g., readability, content-related, etc.). We are also thinking of using bad summaries (e.g., randomly generated summaries from the original texts, or human summaries meant to contain a specific type of error) as negative examples, meant to provide additional information to the model summaries and act as anti-models. This future direction is also in accordance to some interesting findings from the organization of the MultiLing Pilot of TAC2011 (Giannakopoulos et al., 2011): in a language with human models that were graded low, the AutoSummENG method correlated negatively to human grades. This may imply that information on what is bad may be as important to what is good for a given summary and help improve existing measures.

The summarization evaluation domain is just gaining the momentum required to face its challenging research problems. Novel thinking and deep

Group A - All Peers

	Pyramid		Readability		Responsiveness	
	Agreement	Disagreement	Agreement	Disagreement	Agreement	Disagreement
AutoSummENG (12)	195	213	196	212	195	213
MeMoG (18)	408	0	407	1	408	0

Group B - All Peers

	Pyramid		Readability		Responsiveness	
	Agreement	Disagreement	Agreement	Disagreement	Agreement	Disagreement
AutoSummENG (12)	253	155	253	155	253	155
MeMoG (18)	408	0	408	0	408	0

Table 7: Agreement concerning discrimination - All Peers

Group A - No Models

	Pyramid		Readability		Responsiveness	
	Agreement	Disagreement	Agreement	Disagreement	Agreement	Disagreement
AutoSummENG (12)	1061	214	920	355	1039	236
MeMoG (18)	1025	250	900	375	1010	266

Group B - No Models

	Pyramid		Readability		Responsiveness	
	Agreement	Disagreement	Agreement	Disagreement	Agreement	Disagreement
AutoSummENG (12)	1161	114	1015	260	1164	111
MeMoG (18)	1172	103	1032	243	1183	92

Table 8: Agreement concerning discrimination - No Models

study of previous scientific results are needed to improve the performance of evaluators, which will then help determine methods to summarize well.

References

[Banko and Vanderwende2004] Michele Banko and Lucy Vanderwende. 2004. Using n-grams to understand the nature of summaries. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Short Papers*, pages 1–4, Boston, Massachusetts, USA, May. Association for Computational Linguistics.

[Bunke1998] H. Bunke. 1998. Error-tolerant graph matching: a formal framework and algorithms. *Advances in Pattern Recognition, LNCS*, 1451:1–14.

[Copeck and Szpakowicz2004] T. Copeck and S. Szpakowicz. 2004. Vocabulary usage in newswire summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 19–26. Association for Computational Linguistics.

[Dang and Owczarzak2008] H. T. Dang and K. Owczarzak. 2008. Overview of the TAC 2008 update summarization task. In *TAC 2008 Workshop - Notebook papers and results*, pages 10–23, Maryland MD, USA, November.

[Dang2005] H. T. Dang. 2005. Overview of DUC 2005. In *Proceedings of the Document Understanding Conf. Wksp. 2005 (DUC 2005) at the Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*.

[Giannakopoulos and Karkaletsis2010] George Giannakopoulos and Vangelis Karkaletsis. 2010. Summarization system evaluation variations based on n-gram graphs. In *TAC 2010 Workshop*, Maryland MD, USA, November.

[Giannakopoulos et al.2008] George Giannakopoulos, Vangelis Karkaletsis, George Vouros, and Panagiotis Stamatopoulos. 2008. Summarization system evaluation revisited: N-gram graphs. *ACM Trans. Speech Lang. Process.*, 5(3):1–39.

[Giannakopoulos et al.2011] George Giannakopoulos, Benoît Favre, Marina Litvak, Josef Steinberger, and Vasudeva Varma. 2011. TAC 2011 MultiLing pilot overview. In *TAC 2011 Workshop*, Maryland MD, USA, November.

[Giannakopoulos2009] George Giannakopoulos. 2009. *Automatic Summarization from Multiple Documents*. Ph.D. thesis, Department of Information and Communication Systems Engineering, University of the Aegean, Samos, Greece, http://www.iit.demokritos.gr/˜ggianna/thesis.pdf, April.

[Hovy et al.2005a] E. Hovy, C. Y. Lin, and L. Zhou.

2005a. Evaluating duc 2005 using basic elements. *Proceedings of DUC-2005*.

[Hovy et al.2005b] E. Hovy, C. Y. Lin, L. Zhou, and J. Fukumoto. 2005b. Basic elements.

[Lin and Hovy2003] Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 71–78, Morristown, NJ, USA. Association for Computational Linguistics.

[Lin2004] C. Y. Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, pages 25–26.

[Navarro2001] G. Navarro. 2001. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88.

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

[Raymond et al.2002] J. W. Raymond, E. J. Gardiner, and P. Willett. 2002. Rascal: Calculation of graph similarity using maximum common edge subgraphs. *The Computer Journal*, 45(6):631.

[Schilder and Kondadadi2009] F. Schilder and R. Kondadadi, 2009. *A Metric for Automatically Evaluating Coherent Summaries via Context Chains*, pages 65–70.

[Zhou et al.2006] L. Zhou, C. Y. Lin, D. S. Munteanu, and E. Hovy. 2006. Paraeval: Using paraphrases to evaluate summaries automatically. In *Proceedings of the Human Language Technology Conference - North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT/NAACL-2006)*.

A Discriminative Power Definitions

The following example and definitions originate from the README_aesop.txt file in the evaluation results, kindly provided by NIST.

```
                    Pyramid
Submission 7
  224 294   0
  143 820   0
    0   0   4

(Column1,Row1) shows the number of pairs of summarizers (X,Y), where
the AESOP metric (Submission 7) and the Pyramid method agree that
summarizer X is significantly better than summarizer Y, or that
summarizer Y is significantly better than summarizer X (here: 224
agreements).

(Column2,Row1) shows the number of pairs of summarizers (X,Y), where
there is a significant difference between X and Y according to the
AESOP metric, but there is no significant difference according to the
Pyramid method (here: 294 disagreements).

(Column1,Row2) shows the number of pairs of summarizers (X,Y), where
there is a significant difference between X and Y according to the
Pyramid method, but there is no significant difference according to
the AESOP metric (here: 143 disagreements).

(Column2,Row2) shows the number of pairs of summarizers (X,Y), where
the AESOP metric and Pyramid method agree that there is no significant
difference between summarizer X and summarizer Y (here: 820
agreements).

(Column3,Row3) shows the number of pairs of summarizers (X,Y), where
the AESOP metric and the Pyramid method both say that summarizer X is
significantly different from summarizer Y, but disagree as to which
summarizer is better (here: 4 disagreements). Either

    (1) according to the AESOP metric X < Y,
    and according to Pyramid Y < X; or

    (2) according to the AESOP metric Y < X,
    and according to Pyramid X < Y.

The remaining cells can be ignored.
```

In this work, we consider as *Agreements* the sum of cells $(Column1, Row1) + (Column2, Row2)$ and as *Disagreements* the sum of the remaining cells $(Column1, Row2) + (Column2, Row1) + (Column3, Row3)$. The data were provided by NIST.