# N-GRAM GRAPHS: REPRESENTING DOCUMENTS AND DOCUMENT SETS IN SUMMARY SYSTEM EVALUATION

GEORGE GIANNAKOPOULOS
UNIVERSITY OF TRENTO, ITALY AND NCSR DEMOKRITOS, GREECE
AND
VANGELIS KARKALETSIS
NCSR DEMOKRITOS, GREECE

Abstract. Within this article, we present the application of the AutoSummENG method within the TAC 2009 AESOP challenge. We further offer an alternative to the original AutoSummENG method, which uses an additional operator of the n-gram graph framework to represent a set of documents with a single, merged graph. Both alternatives offer very good results in different aspects of the AESOP results evaluation. The original AutoSummENG method appears a very good linear estimator of Pyramid score and responsiveness, while the new Merged Model variation offers very good (non-linear) rank estimation performance when correlated to the responsiveness measure.

## 1. Introduction

The problem of automatically determining the quality of a given summary appears to be approached using two different perspectives in the current literature: either by *intrinsic* or *extrinsic* evaluation [MB97, VHT03]. Intrinsic evaluation operates on the characteristics of the summary itself, independent of the domain it may be used, trying for example to capture how many of the ideas expressed in the original sources appear in the output. On the other hand, extrinsic evaluation decides upon the quality of a summary depending on the effectiveness of using the latter for a specified task. Such a measure of extrinsic evaluation, namely *responsiveness*, appeared in the Document Understanding Conference (DUC) of 2005[1]. This extrinsic measure has been used in later DUCs and Text Analysis Conferences (TACs) as well.

Sparck Jones in [Jon07] argues that the classification of evaluation methods as intrinsic and extrinsic is not enough and proposes an alternative schema of evaluation methods' classification. This schema is based on the degree to which the evaluation method measures performance, according to the intended purpose of the summary. Therefore, defining new classes that elaborate on the definitions of extrinsic and intrinsic, Sparck Jones classifies evaluation methodologies as:

- semi-purpose, *e.g.* inspection of proper English.
- quasi-purpose, based on comparison with models, *e.g.* n-gram or information nuggets.
- pseudo-purpose, based on the simulation of task contexts, *e.g.* action scenarios.
- full-purpose, based on summary operation in actual context, *e.g.* report writing.

The higher in the above list an evaluation method is mapped, the more it appeals to the notion of intrinsic, while the lower it maps the more it would be considered extrinsic.

[1] Also see *http://duc.nist.gov/*

In [BDH+00] we find a comment (part 3.4) referring to intrinsic evaluation, where the authors suggest that 'only humans can reliably assess the readability and coherence of texts'. This statement indicates the difficulty of that kind of evaluation. There appears to be no absolute measure of quality for a summary, even for human judges. Thus, an automatic measurement of the quality of a summary would require at least one *model summary* (*i.e.* human extracted summary produced as a reference for measuring the goodness of the summaries produced by others), also called 'gold standard' or 'reference' summary. The human summaries offer high responsiveness content. This given, it would be possible to judge the *peer summaries* (*i.e.* summaries extracted by peer systems). Such measurements actually determine some kind of distance between the peer and the model summaries. Within this work we present an intrinsic method of evaluation based on the existence of model summaries.

Automatic methods for the evaluation of summaries exist [HLZF05, Lin04, ZLMH06] and correlate highly to the measure of responsiveness. Until recently, however, there were some desired characteristics that did not coexist in a single method. More precisely:

- Language-neutrality. A method that does not require language dependent resources (thesauri, lexica, etc.) can be applied directly to different languages.
- 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.

Our method, named AutoSummENG[GKVS08] (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. The results of our experiments on the TAC2009 corpus indicated that our method still holds state-of-the-art performance in that sort of correlation, while remaining strictly statistical, automated and context-sensitive due to the nature of the representation used, namely the n-gram graph.

## 2. System Overview

The AutoSummENG system[GKVS08] is based upon the JInsect library[2] of processing text with use of the n-gram graph representation. The performed experiments this year simply applied the same evaluation method over the TAC 2009 AESOP task data, under two different variations:

- The first is the original AutoSummENG, for detected optimal parameters of $L_{\mathrm{min}}$, $L_{\mathrm{MAX}}$ and $D_{\mathrm{win}}$ 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 the overall performance of the summary text.
- The second variation, 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.

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

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

2.1. **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 2.1). N-grams have been used in summarization and summary evaluation [BV04, LH03, CS04]. In the automatic summarization domain, n-grams appear as word n-grams, as happens in the ROUGE/BE family of evaluator methods [HLZ05, Lin04].

**Example 2.1.** 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_, ...*

2.2. **Extracting N-grams.** If we choose to extract the n-grams ($S^n$) of a text $T^l$, the (elementary) algorithm is 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
$SS^n \leftarrow \emptyset$;
for all i in [1,length(T)-n+1] do
    $SS^n \leftarrow SS^n \cup S_{i,i+n-1}$
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 if you do not know the exact center of the match between two subparts of a string. In the case of summary evaluation against a model summary for example, the extracted n-grams are certain to include n-grams of the model summary, if such an n-gram exists, whereas a method where the text would be segmented in equally sized n-grams might not identify similar n-grams.

**Example 2.2.** *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 *n-gram graph* is a graph $G = \{V^G, E^G, L, W\}$, where $V^G$ is the set of vertices, $E^G$ 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 $v^G \in V^G$ and 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 can be weighted by the distance between the two neighboring n-grams in the original text, or the number of co-occurrences within a given window (as indicated below we use the co-occurrences for the TAC task). We note that the meaning of distance and window size changes by whether we use character or word n-grams. 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 2.3.** *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 $D_{\text{win}}$ of each other. We take into account only a window around $S_i$ in the original text, to determine which $S_j$ is useful. The vertices $v_i, v_j$ corresponding to n-grams $S_i, S_j$ that are located within this parameter distance $D_{\text{win}}$ are connected by a corresponding edge $e \equiv \{v_i, v_j\}$.

In the TAC 2009 case we use the *symmetric approach* for *character n-gram* graph extraction, which has proved to be the most promising in several experiments [GKVS08].

## 3. Graph Matching

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 [Bun98].

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 [RGW02]).

**Edit-distance Based:** Edit distance has been used in fuzzy string matching for some time now, using many variations (see [Nav01] 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 [Bun98]. 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. A graph method for text comparison can be found in [TNI04], where a text is represented by first determining weights for the text's terms using a TF-IDF calculation and then by creating graph edges based on the term co-occurrences. In our method, no term extraction is required and the graph is based directly on the text, without further background such as a corpus for the calculation of TF-IDF or any other weighting factor. We have applied our own graph matching method, that 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 documents with a single graph, as we show in the following sections.

## 4. Representing Document Sets

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. For example, let the merging of $G_1$ and $G_2$ corresponding to the union operator in set theory be $G_3 = G_1 \cup G_2$, which is implemented by adding all edges from both graphs to a third one, while making sure no duplicate edges are created. Two edges are considered duplicates of each other, when they share identical vertices[3].

The invention of the operator is actually non-trivial, because a number of questions arise, such as the handling of weights on common edges after a union operation or the keeping of zero-weighted edges after the application of an operator. In our implementation we have decided that the union operator will average existing edge weights for every common edge into the corresponding new graph edge. Zero-weighted edges are treated like all other edges, even though they have the same semantics as the absence of an edge (*i.e.* the vertices are not related).

In all the presented algorithms we work with edges only, because the way the graphs have been created does not allow isolated vertices to exist. Throughout this section we consider that information is contained within the relations between n-grams and not in the n-grams themselves. Therefore, our minimal unit of interest is the edge, which is actually a pair of vertices. This use of graphs implicitly defines the properties of the graph's vertices, based on what we do with the corresponding edges.

The merging or union operator $\cup$ for two graphs, returning a graph with all the edges, both common and uncommon, of the two operand graphs. The edges are weighted by the average of the weights of the original edges. The intuition behind averaging the edges' weights is that the merged graphs should be equally close to the two operand graphs in terms of edge weights; this is the effect of an averaging function.

In our applications we have used an *update function* $U$ that is similar to the merging operator, with the exception that the weights of the resulting graph's edges are calculated in a different way. 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

[3] The identity between vertices can be a customized calculation. Within our applications two vertices are the same if they refer to the same n-gram, *i.e.* they share the same label. Thus, identity can be checked by simple string matching.

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 simple merging operator. The definition of the weighting performed in the graph resulting from $U$ is:

$$W^i(e) = W^1(e) + (W^2(e) - W^1(e)) \times l \tag{1}$$

The $U$ function allows using the graph 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 $l = 1.0 - \frac{i-1}{i}, i > 1$. This gives a graph whose role is similar to the centroid of a set of vectors: it functions as a representative graph for the set its constituent graphs.

## 5. Comparison

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 $\frac{\mathrm{VR}(e)}{\max(|G^i|,|G^j|)}$ 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:

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

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

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

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

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

The fraction $\mathrm{SS}(G^i, G^j) = \frac{\min(|G^i|,|G^j|)}{\max(|G^i|,|G^j|)}$, 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 2009 case there is no real difference, however, because the SS factor is almost constant and equal to 1, because the summaries have an almost fixed size.

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

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

where $\mathrm{VS}^r$ is the VS measure for extracted graphs of rank $r$ in $\mathbb{G}$, and $L_{\min}$, $L_{\mathrm{MAX}}$ 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 $O(h\mathrm{min}(|G_1|, |G_2|))$, 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 $G_2$.

## 6. Experiments

The experiments conducted upon the TAC 2009 corpus were based on the application of the AutoSummENG on the TAC corpus, using either the individual model summaries' graphs or their merged graph as a model. The parameters of the evaluation method include the minimum and maximum character n-gram sizes taken into account, as well as the maximum distance between n-gram taken into consideration to form the edges between neighboring n-grams. These parameters are derived from an a priori parameter estimation process that separates n-grams into meaningful ones, called *symbols*, and useless ones, called *non-symbols*. The distinction between symbols and non-symbols is based on statistical measures (see [GKVS08] for more on symbols, non-symbols and parameter estimation) and, as such, is language independent. On the TAC 2009 corpus the proposed parameters were $(L_{\min}, L_{\mathrm{MAX}}, D_{\mathrm{win}}) = (3, 3, 3)$, which seems to stand for many English corpora.

The summaries in the AESOP test data consist of all the model summaries and "automatic" (non-model) summaries produced within the TAC 2009 Update Summarization task. 8 human summarizers produced a total of 352 model summaries, and 55 "automatic" summarizers produced a total of 4840 "automatic" summaries. The set of systems included three baseline summarizers:

- Summarizer 1: returns all the leading sentences (up to 100 words) in the most recent document. Summarizer 1 provides a lower bound on what can be achieved with a simple fully automatic extractive summarizer.
- Summarizer 2: returns a copy of one of the model summaries for the docset, but with the sentences randomly ordered. Summarizer 2 provides a way of testing the effect of poor linguistic quality on the overall responsiveness of an otherwise good abstractive summary.
- Summarizer 3: returns a summary consisting of sentences that have been manually selected from the docset. Summarizer 3 provides an approximate upper bound on what can be achieved with a purely extractive summarizer. [4].

[4] This HexTac summarizer (Human EXtraction for TAC) was contributed by a team of five human "sentence-extractors" from the University of Montreal.

| RunID | Pyramid | | | Responsiveness | | |
|---|---|---|---|---|---|---|
| | Pearson | Spearman | Kendall | Pearson | Spearman | Kendall |
| | Main Summaries | | | | | |
| 11 | 0.982 **(2)** | 0.953 **(5)** | 0.826 **(3)** | 0.968 **(1)** | 0.894 **(5)** | 0.735 **(5)** |
| 31 | 0.966 **(5)** | 0.938 (12) | 0.795 (14) | 0.958 **(3)** | 0.913 **(1)** | 0.761 **(1)** |
| *Min* | -0.349 | 0.038 | 0.092 | -0.407 | -0.019 | 0.040 |
| *Mean* | 0.682 | 0.826 | 0.688 | 0.606 | 0.769 | 0.615 |
| *St. Dev.* | 0.286 | 0.216 | 0.197 | 0.299 | 0.209 | 0.177 |
| *Max* | 0.983 | 0.962 | 0.847 | 0.968 | 0.913 | 0.761 |
| | Update Summaries | | | | | |
| 11 | 0.976 **(2)** | 0.942 **(4)** | 0.807 (7) | 0.963 **(1)** | 0.851 (10) | 0.702 (8) |
| 31 | 0.963 **(5)** | 0.920 (15) | 0.777 (16) | 0.957 **(3)** | 0.875 **(3)** | 0.728 **(1)** |
| *Min* | -0.353 | 0.096 | 0.119 | -0.404 | 0.006 | 0.071 |
| *Mean* | 0.642 | 0.794 | 0.664 | 0.548 | 0.708 | 0.568 |
| *St. Dev.* | 0.325 | 0.249 | 0.222 | 0.325 | 0.241 | 0.201 |
| *Max* | 0.978 | 0.966 | 0.858 | 0.963 | 0.878 | 0.728 |

Table 1. Correlation-based Performance of traditional AutoSummENG (RunID 11) and merged model AutoSummENG (RunID 31) over *all summaries*. For all the AutoSummENG correlation tests the p-value is $< 0.001$. Ranks within the top 5 positions indicated by **bold** font.

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

The summary of the performance of the AutoSummENG alternatives to the Pyramid and Responsiveness measures is depicted in Table 1. According to the organizers, in the AESOP task, two baseline automatic metrics were included in the evaluation: System 1: ROUGE-SU4, with stemming and keeping stopwords and System 2: Basic Elements (BE)[6]. An additional 35 metrics were submitted by 12 participants in the AESOP task, resulting in 37 AESOP metrics that were evaluated. In Table 1 for each performance-indicative column we also show the maximum, the minimum, the mean and the standard deviation of the corresponding correlation over all the 37 metrics. The values correspond to the performance over all evaluated summaries (human-composed and automatically extracted). In parentheses we indicate the rank of the AutoSummENG variations based on each correlation. We note that Pearson correlation indicates linear correlation, while Spearman and Kendall are rank-based, thus not caring about whether correlation is linear or not.

What is really interesting is the fact that the Merged Model alternative is maximally correlated (ranks 1st amidst the 37 systems) to the human metric of responsiveness using Kendall's Tau. This indicates that the merged model holds all the necessary information that can support the responsiveness quality of a given text. In other words, this averaged view of the model documents, when used in conjunction with the similarity we have devised, appears to provide ranking in a way very similar to humans, when evaluating summary responsiveness. On the other hand, the original AutoSummENG method still holds very high Pearson correlation to, *i.e.* is a very good linear estimator of, the Pyramid score (see bold indicated ranks in Table 1) and Responsiveness.

[5] See http://www.nist.gov/tac for more info on the Summarization Task of TAC 2009.

[6] Summaries were parsed with Minipar, and BE-F were extracted and matched using the Head-Modifier criterion.

| RunID | Pyramid | | | Responsiveness | | |
|---|---|---|---|---|---|---|
| | Pearson | Spearman | Kendall | Pearson | Spearman | Kendall |
| | Main Summaries | | | | | |
| 11 | 0.954 (7) | 0.933 (6) | 0.796 **(5)** | 0.829 (9) | 0.843 (8) | 0.669 (9) |
| 31 | 0.894 (17) | 0.912 (16) | 0.758 (17) | 0.756 (22) | 0.873 **(1)** | 0.707 **(1)** |
| | Update Summaries | | | | | |
| 11 | 0.970 **(1)** | 0.918 (10) | 0.772 (12) | 0.796 (8) | 0.778 (12) | 0.625 (11) |
| 31 | 0.940 (11) | 0.884 (18) | 0.734 (18) | 0.741 (14) | 0.815 **(5)** | 0.657 **(5)** |

Table 2. Correlation-based Performance of traditional AutoSummENG (RunID 11) and merged model AutoSummENG (RunID 31) over *non-model peers*. For all the AutoSummENG correlation tests the p-value is $< 0.001$. Ranks within the top 5 positions indicated by **bold** font.

The system performance is slightly reranked when model summaries are not taken into account, essentially judging only automatic summaries (see Table 2). However, the AutoSummENG performance still remains within the top ranks amidst the 37 peers. Especially the Merged Model version remains 1st in its Kendall correlation to responsiveness for group A of summaries. In group B (update summaries) it ranks 5th[7].

## 7. Conclusions - Future work

Our experiments on the TAC2009 corpus indicated that:

- the AutoSummENG is robust and performs well as a system evaluation measure for multi-document summaries given model summaries, when used to estimate responsiveness-based ranking or to estimate Pyramid method values through linear estimation.
- the merging of model texts' representations into an overall model graph can function as an "average document graph", that is representative of the model summaries and performs really well by means of correlation to responsiveness.
- the original AutoSummENG method is mostly linearly correlated to the Pyramid evaluation, while the *Merged Model* variation is very correlated to the (human) responsiveness metric (though not through linear correlation).
- A completely statistical summarization system evaluation method can perform at the state-of-the-art level, without any need for preprocessing or deep analysis.

It would be important to be able to combine well-performing methods from the set of methods used in the TAC task, using the ones that are mostly uncorrelated to each other. This practice would aim to combine somewhat weaker estimators which are maximally uncorrelated into a single powerful estimator of performance. This however requires the collaboration of different research groups under a common cause.

In the future work, we plan to:

- Elaborate further on the modeling ability of the update/merge operator, depending on the number of constituent texts.

[7] The best Kendall's Tau-based Responsiveness correlation performance, by system 15, in the no-models, update summaries case is less 0.02 above the AutoSummENG MM variation performance. This stands among several cases in the top ranks. Statistical hypothesis testing should be further used to indicate groups of systems with significantly important difference in performance.

- Try to remove the noise from the graphs before performing similarity measurements (see [Gia09] for more information on how noise can be defined in the graph representation of texts).
- Use a multilingual corpus of summaries to determine the performance of AutoSummENG on non-English texts. The optimal parameters also will be an interesting case study.

## References

[BDH$^+$00] B. Baldwin, R. Donaway, E. Hovy, E. Liddy, I. Mani, D. Marcu, K. McKeown, V. Mittal, M. Moens, D. Radev, and Others. An evaluation roadmap for summarization research. Technical report, 2000.

[Bun98] H. Bunke. Error-tolerant graph matching: a formal framework and algorithms. *Advances in Pattern Recognition, LNCS*, 1451:1–14, 1998.

[BV04] Michele Banko and Lucy Vanderwende. 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 2004. Association for Computational Linguistics.

[CS04] T. Copeck and S. Szpakowicz. Vocabulary usage in newswire summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 19–26. Association for Computational Linguistics, 2004.

[Gia09] George Giannakopoulos. *Automatic Summarization from Multiple Documents*. PhD thesis, Department of Information and Communication Systems Engineering, University of the Aegean, Samos, Greece, http://www.iit.demokritos.gr/~ggianna/thesis.pdf, April 2009.

[GKVS08] George Giannakopoulos, Vangelis Karkaletsis, George Vouros, and Panagiotis Stamatopoulos. Summarization system evaluation revisited: N-gram graphs. *ACM Trans. Speech Lang. Process.*, 5(3):1–39, 2008.

[HLZ05] E. Hovy, C. Y. Lin, and L. Zhou. Evaluating duc 2005 using basic elements. *Proceedings of DUC-2005*, 2005.

[HLZF05] E. Hovy, C. Y. Lin, L. Zhou, and J. Fukumoto. Basic elements, 2005.

[Jon07] Karen Sparck Jones. Automatic summarising: The state of the art. *Information Processing & Management*, 43(6):1449 – 1481, 2007. Text Summarization.

[LH03] Chin-Yew Lin and Eduard Hovy. 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, 2003. Association for Computational Linguistics.

[Lin04] C. Y. Lin. Rouge: A package for automatic evaluation of summaries. *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, pages 25–26, 2004.

[MB97] Inderjeet Mani and Eric Bloedorn. Multi-document summarization by graph search and matching. In *Proceedings of AAAI-97*, pages 622–628. AAAI, 1997.

[Nav01] G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.

[RGW02] J. W. Raymond, E. J. Gardiner, and P. Willett. Rascal: Calculation of graph similarity using maximum common edge subgraphs. *The Computer Journal*, 45(6):631, 2002.

[TNI04] Junji Tomita, Hidekazu Nakawatase, and Megumi Ishii. Calculating similarity between texts using graph-based text representation model. pages 248–249, Washington, D.C., USA, 2004. ACM.

[VHT03] H. Van Halteren and S. Teufel. Examining the consensus between human summaries: Initial experiments with factoid analysis. In *Proceedings of the HLT-NAACL 03 on Text Summarization Workshop-Volume 5*, pages 57–64. Association for Computational Linguistics Morristown, NJ, USA, 2003.

[ZLMH06] L. Zhou, C. Y. Lin, D. S. Munteanu, and E. Hovy. 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)*, 2006.