Rich Semantic Graph: A New Semantic Text Representation
Approach for Arabic Language

Sally S. Ismail
Computer Science Dept.
Faculty of Computer and Information Sciences, Ain Shams University, Abbassia, Cairo, Egypt
sallysaad@cis.asu.edu.eg

Mostafa Aref
Computer Science Dept.
aref_99@yahoo.com

Ibrahim Moawad
Information Systems Dept.
ibrahim_moawad@cis.asu.edu.eg

Abstract: - Arabic Text Summarization is one of the challenging open areas for research in Natural Language Processing (NLP) field. Representing Arabic text semantically can facilitate this process by helping in understanding the highly complicated semantic structure of the Arabic language. The work presented in this paper is a part of an ongoing research to create an abstractive summary for a single input document in the Arabic Language. The abstractive Summary is generated through three modules; converting the input Arabic text into a semantic graph called Rich Semantic Graph (RSG), then reducing the created RSG, and finally generating the summary from the reduced graph. In a previous research, we have proposed that model for Arabic Text Summarization, and in this paper we are presenting the first module of this model.

Key-Words: - Arabic Natural Language Processing, Text Summarization, Ontology, Rich Semantic Graph.

1 Introduction

In this era, while information is becoming more and more accessible, it is impossible to read all the relevant content that helps us stay informed. A possible solution would be condensing data and obtaining the kernel of a text to explore, analyze, and discover knowledge from documents. For this reason, text summarization has quickly grown into a major research area as illustrated by the Document Understanding Conference (DUC) and Text Analysis Conference (TAC) series [1,2]. Text Summarization techniques can be classified according to the number of input documents (single-document versus multi-document), the documents type (textual versus multimedia), or the output type (extractive versus abstractive).

Text Summarization systems for Arabic are still not as advanced and as reliable as those developed for other languages like English. Till now, Arabic Natural Language Processing (NLP) has focused on the manipulation and processing of the structure of the language at morphological, lexical, and syntactic levels. Unfortunately, semantic processing of the Arabic language has not yet received enough attention. And although it is a very rich language, most of the studies consider its derivational and inflectional aspects as a disadvantage, rather than an advantage. Thus, the specifications of the Arabic language were barely utilized, instead, approaches that were successful with roman languages such as English and French were applied on it. Some of the aspects that slow down progress in Arabic NLP include its complex morphology, the absence of diacritics in written text and the fact that Arabic does not use capitalization. In addition, there is also a shortage of Arabic corpora, lexicons, Ontologies and machine-readable dictionaries. These tools are essential to advance research in different areas. A domain-specific Ontology, for example, is essential for many NLP applications such as Information Retrieval, Question Answering Systems, Natural Language Generation, Machine Translation, and Text Summarization. Thus, utilizing an Ontology in representing text in a Semantic Graph can facilitate the process of getting the semantics behind the Arabic text.

Our work in this paper is a part of an ongoing research on Ontology based approach for Arabic Text Summarization. In an earlier work, we have proposed a model where a single document text is represented in a Rich Semantic Graph, then the RSG is reduced, and finally Sentence generation model is invoked to produce an abstractive summary [3]. All this is done with the aid of an Arabic domain specific Ontology, which helps in concepts descriptions, relations and generalization.
In this paper, we are going to discuss the first module of the suggested model, which is Text to RSG. In section 2, some of the related work for this area is to be declared, while section 3 gives an overview on the proposed model. Section 4 illustrates the proposed design for Text to RSG Module, and then a test case is examined in section 5. We finalize the paper with the conclusion as well as the future work in section 6.

2. Related Work

Several research activities are related to Graph Representation for Text summarization. While most of the earlier work stays at the shallow parsing level, in 2004, Leskovec et al[4-6] presented a novel approach of document Abstractive summarization by generating semantic representation of the document; which introduces an intermediate, more generic layer of text representation within which the structure and content of the document and summary are captured. The intermediate semantic graph representation opens up new areas of explorations. They then apply machine learning to extract a sub-graph that corresponds to the semantic structure of a human extracted document summary using features that capture semantic structure, i.e. concepts and relations, in contrast to previous attempts in which linguistic features are of finer granularity, i.e. keywords and noun phrases. This direction of research has been outlined by (Spark Jones) in [7] as still unexplored avenue.

One of the most recent research methods is extracting summary sentences based on the document semantic graph representation. This method generates semantic representation from the input document by a machine learning technique to extract full sentences suitable for creating summaries. It starts with deep syntactic analysis of the whole text, then for each sentence extracts logical form triples (subject-predicate-object). After that, it applies cross sentence pronoun resolution, co-reference resolution, and semantic normalization to refine the set of triples and merge them into a semantic graph. This procedure is applied to both document and corresponding summary extracts. Finally, linear support vector machine is trained on the logical form triples to learn how to extract triples that belong to sentences in document summaries. The classifier is then used for automatic creation of document summaries of test documents [8,9].

On the other hand, there's an approach that incorporates the object-orientation techniques in knowledge representation. It supports data abstraction and hence increases the modularity of natural language processing applications. It organizes knowledge into classes of objects (subclasses and super classes), which is an important issue in knowledge representation to avoid redundant declarations or specifications. [10].

However, semantic graph representation in its traditional form is incomplete due to the limitation of explicit operational or procedural knowledge, so it needs to assign more structure to nodes as well as to links. On the other hand, object-oriented (OO) modeling reflects the structure of data and the software's object behavior not the world concepts and its structure. Therefore, only traditional object-oriented technique is not enough for a good knowledge representation. It is difficult to build large coherent and complete knowledge representation. Finally, the granularity is an important issue in knowledge representation. It concerns with the level of knowledge details that should be represented and what are the primitives (fundamental concepts) needed to be constructed.

Moawad et al. [11-13] followed the idea of graph representation but using Ontology rather than Machine Learning techniques to perform the Abstraction. They designed a system model for Abstractive Summarization in English Text. They also implemented a prototype to transform an input document to a Rich Semantic Graph (RSG) which is based on ontology. The ontology primitives (concepts) are language nouns and verbs only. It preserves words hierarchies and their semantic constraints and ontological relations among each other. Therefore, the output graph is not complex, and not huge, but rich, because each concept has its own linguistic and semantic attributes and relations that can be deduced from the analyzed input text.

As for the Arabic Text Summarization, some efforts have been performed such as; the Optimized Dual Classification System [14], which represents an Arabic Extractive text summarization system. Based on the Rhetorical Structure Theory (RST), AlSanie [15] developed an Arabic text summarization system, and Ikhtasir [16] also is an integrated RST-based system with a certain scoring scheme. On the
other hand, there are systems which summarize multi-lingual sets of documents like Lakhas [17] and CLASSY (Clustering, Linguistics, And Statistics for Summarization Yield) in [18].

3. Arabic Text Summarization Model

None of the aforementioned Arabic Text summarization systems utilize an Ontology for doing abstractive summary. Thus, in the proposed model we are going on the track of Moawad et al.’s design[11-13] which utilizes a domain Ontology for creating an abstractive summary for a single Arabic input document. We are guided by Sparck Jones research[7], who formerly decomposed summarization into three main phases: Text Representation, Transformation of the text representation into a summary representation, and Generation of the summary text from the summary representation. We are putting into consideration the Arabic language characteristics. This approach is considered a novel one for the Arabic Language

In a previous research [3], we proposed the model design, which consists of three modules: creating a semantic graph for the original document, reducing the generated semantic graph to an abstracted semantic graph, and applying a natural language generation technique on the semantic graph to generate the abstractive summary as depicted in Fig.1.

![Fig 1. Overview on the proposed design](image)

In this paper, we are presenting the first Module; representation of Text in a RSG. The main objective of this Module is to represent the input document semantically using a semantic graph, where words and concepts of the input document are represented as instance objects of the corresponding word classes in the graph along with edges corresponding to semantic and topological relations between them. This representation is called Rich Semantic Graph, because it is able to capture the meaning of words, sentences and paragraphs. Each node is rich with its attributes, which include: the value of the node, type(noun or verb), descriptors, and attachments. For example, if the concept in the node is for a verb then the attributes include its subject(s), object(s), tense, place...etc.

4. Arabic Text to RSG Module

Text to RSG module is accomplished through 5 phases:
1. Preprocessing.
2. Word Sense Instantiation.
3. Concept Validation.
4. Sentence Ranking.
5. RSG Generation.

In Preprocessing phase, the document text is read, POS Tagged and Named entities are recognized. Then analysis for the text is done morphologically and syntactically. Finally, co-reference and pronominal resolution is performed on the given text.

As for Word Sense Instantiation phase, instantiate a concept node for each aforementioned tagged noun or verb, , and fetch all their senses in the Ontology. Reduce number of found senses by considering only the sense with the matched type (v/n) of the concept labeled in the POS Tagging process.

Concepts are interconnected and validated through semantic and syntactic constraints and relationships using Ontology and pre-processed Tags, in Concept Validation phase. Ontology is checked to confirm combination consistency for concepts. Inconsistent combinations are rejected. The most commonly used senses are given higher priority.

As for Sentence Ranking phase, we start on the Concept level, by ranking the validated senses according to their relevance from 1→n (Assumption: the more preceding position/frequently used in the Ontology a sense takes the more relevant it is assumed to be). Then, we Normalize every sense Rank (SRi,j) to be in the range of 0→10 using Equation 1: (with value 10 to be the most significant and 0 to be least significant)
\[ SR_{i,j} = \frac{n - j + 1}{n} \quad \text{Eq.1} \]

where, \( SR_{i,j} \) represents the rank of the sense number \( j \) for Concept number \( i \), \( n \) stands for the Total number of valid senses for this concept.

Then the Ranks are thresholded to take the highest valid senses. By analyzing the ranks we chose to take the threshold at value 6.

For each sentence, compute the number of possible Combinations of all the senses of all of its concepts before thresholding (this is the original number of RS Sub-Gs) and after thresholding the senses (a reduced no of RS Sub-Gs). \( \text{Calculate Average Sentence Rank} \) for all possible combinations between all concepts using \( \text{Equation 2} \).

\[ ASR_k = \frac{\sum_{i=1}^{N} SR_{i,j}}{N} \quad \text{Eq.2} \]

where, \( ASR_k \) stands for Average Ranking number \( k \) for the sentence, and \( N \) is the Total number of concepts in the sentence.

Afterwards, a Threshold is taken on the Average Sentence Ranks of each sentence to reach the nominated \( n \) sentences to be considered in RS Sub-Graphing. By analyzing the Sentence ranks we chose to take the threshold at value 9.5.

Finally, RSG Generation phase works both on the sentence and the paragraph levels. On the Sentence Level, we specify Validated Senses for each concept, then Verify the Relations between different concepts (Consulting the Ontology). Non-matching senses are excluded, and finally Rich Semantic Sub Graph(s) connecting the remaining senses together are generated.

Similarly, on the Paragraph Level, we verify accepted sentences/sub-sentences together. Relations between them is specified through the Ontology and through formerly determined Coreference and Pronominal Resolutions. Finally, connect accepted RS Sub-Gs to form the complete RSG(s) for the whole paragraph.

5. Case Study

To verify our design for Text Representation, we have used a test case; shown in Fig.2; consisting of 1 Paragraph, divided into 3 main sentences, which are subdivided into 6 sub-sentences, and total number of words (including stop words) is 37.

5.1 Test Case

For simplification, only sub-sentence 1.1 is used to illustrate how we applied the phases of Word Sense Instantiation, Concept Validation, Sentence Ranking, RSG Generation.

After text of this sentence is preprocessed, senses for each verb or noun are extracted from Ontology in Word Sense Instantiation phase. For the considered sub-sentence the original number of senses would have given 8736 combinations for RS Sub-G's and this number is reduced to be 2704 by reviewing the Tags of the words and only considering the matching types of senses.

Then the remaining senses are validated in Concept Validation phase. For example, for verb "رفاق" which we currently have 4 senses:  

- "رفاق" عامل بلطف، صبب معروفه صاحب السر، ضرب بمرفقه.  
- "رفاق" يعجب.  

The first two are accepted, while the 3rd sense is rejected due to lack of preposition in the original text, and the 4th sense is rejected too due to its requirement for another word which is not found in the original text. Again number of senses is reduced to give only 64 RS Sub-G's. Afterwards, Sentence is Ranked as follows: On Concept level; applying Equation 1 on sub-sentence 1.1 Concept # 1:

[ given: \( i=1, Ci=رفاق , n = 2 \)]

- \( j = 1 \)  
  \[ SR_{1,1} = \frac{(2 - 1 + 1)}{2} \times 10 = 10 \]
- \( j = 2 \)  
  \[ SR_{1,2} = \frac{(2 - 2 + 1)}{2} \times 10 = 5 \]

By thresholding at value 6, we exclude Sense \( SR_{1,2} \). Equivalently, other Senses Ranks are mapped and
thresholded. Then on **Sentence Level**: we calculate the number of Combinations of RS Sub-Gs, to sentence 1.1 resulting in **four** different combinations as follows:

\[
given n_{i1} = 1, n_{i2} = 2, n_{i3} = 1, and n_{i4} = 2
\]

\[Cmb = 1 \times 2 \times 1 \times 2 = 4\]

where \(n_{i1}\) is the total number of valid senses for concept number 1.

The four combinations and their computed Average Sentence Ranks (using Equation 2) are:

1. \([10,10,10,10]\) \(\Rightarrow\) \(ASR1 = \frac{(10 + 10 + 10 + 10)}{4} = 10\).
2. \([10,7.5,10,10]\) \(\Rightarrow\) \(ASR2 = \frac{(10 + 7.5 + 10 + 10)}{4} = 9.38\).
3. \([10,10,7.5,10]\) \(\Rightarrow\) \(ASR3 = \frac{(10 + 10 + 7.5 + 10)}{4} = 9.38\).
4. \([10,7.5,10,7.5]\) \(\Rightarrow\) \(ASR4 = \frac{(10 + 10 + 7.5 + 10)}{4} = 8.75\).

By thresholding at value 9.5, we exclude 2.3 and 4. Similarly, other Average Sentences Ranks (ASR's) are calculated and thresholded reducing the number of RS Sub-Graphs for each sentence. Then RS Sub-Gs are drawn for each validated sub-sentence. By far, for this sub-sentence we have reached only one RS Sub-G shown in Fig.3.

On the Paragraph level, relating all validated sub-sentences together and connecting all RS Sub-Gs for the 6 sub-sentences.

Table 1 shows the resulting number of RS Sub-Gs for each sub-sentence and the effect of the aforementioned phases on reducing this number. It also displays the Total number of RS Sub-G’s for all the 6 sub-sentences in all phases.

By multiplying the 6 final numbers of RS Sub-Gs (1.3,1,2,1,3) we’ll be having **18** different complete RSGs for the whole paragraph, one of which is depicted in figure 4.

<table>
<thead>
<tr>
<th>Sentence</th>
<th># of RS Sub-G's</th>
<th>Instantiation</th>
<th>Validation</th>
<th>Ranking</th>
<th>Generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>8,736</td>
<td>2704</td>
<td>64</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>21,840</td>
<td>5040</td>
<td>320</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>2.1</td>
<td>84</td>
<td>84</td>
<td>27</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2.2</td>
<td>7,840</td>
<td>5096</td>
<td>208</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>2.3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>320</td>
<td>104</td>
<td>52</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Total</td>
<td>38,821</td>
<td>13,029</td>
<td>672</td>
<td>14</td>
<td>11</td>
</tr>
</tbody>
</table>

![Fig.3 RS Sub-G for sub-sentence 1.1](image)

![Fig.4 A complete RSG for the whole paragraph](image)

5. **Conclusion**

In this paper, we have proposed a design for **Text to RSG module**, which is one of 3 modules in an ongoing research about Abstractive Arabic Text Summarization System, using a domain-specific Ontology. We have applied the proposed module on a case study as a prototype, to illustrate the results of each phase.
It is considered a novel approach for Arabic Text Summarization; since all previous work in Arabic text do Extractive Summarization and none of them proposed to utilize Ontology neither do they use Rich Semantic Graph.

In our ongoing research, we are going to implement a prototype for the proposed design. To do so we should make use of an existing Arabic domain Ontology.

References