Summarizing Reports on Evolving Events; Part I: Linear Evolution

Stergos D. Afantenos and **Vangelis Karkaletsis**
National Center for Scientific Research (NCSR) "Demokritos", Greece
{stergos,vangelis}@iit.demokritos.gr

Panagiotis Stamatopoulos
Department of Informatics, University of Athens, Greece
takis@di.uoa.gr

Abstract

We present an approach for summarization from multiple documents which report on events that evolve through time, taking into account the different document sources. We distinguish the evolution of an event into *linear* and *non-linear*. According to our approach, each document is represented by a collection of messages which are then used in order to instantiate the cross-document relations that determine the summary content. The paper presents the summarization system that implements this approach through a case study on linear evolution.

1 Introduction

With the advent of the Internet, access to many sources of information has now become much more easier. One problem that arises though from this fact is that of the information overflow. Imagine, for example, that someone wants to keep track of an event that is being described on various news sources, over the Internet, as it evolves through time. The problem is that there exist a plethora of news sources that it becomes very difficult for someone to compare the different versions of the story in each source. Furthermore, the Internet has made it possible now to have a rapid report of the news, almost immediately after they become available. Thus, in many situations it is extremely difficult to follow the rate with which the news are being reported. In such cases, a text summarizing the reports from various sources on the same event, would be handy. In this paper we are concerned with the automatic creation of summaries from multiple documents which describe an event that evolves through time. Such a collection of documents usually contains news reports from various sources, each of which provides novel information on the event as it evolves through time. In many cases the sources will agree on the events that they report and in some others they will adopt a different viewpoint presenting a slightly different version of the events or possibly disagreeing with each other. Such a collection of documents can, for example, be the result of a Topic Detection and Tracking system (Allan *et al.* 98).

The identification of similarities and differences between the documents is a major aspect in Multi-document Summarization (Mani 01; Afantenos *et al.* 05a; Afantenos *et al.* 05b). (Mani & Bloedorn 99), for example, identify similarities and differences among *pairs* of isolated documents by comparing the graphs that they derive from each document, which are based heavily on various lexical criteria. Our approach, in contrast, does not take into consideration isolated pairs of documents, but instead tries to identify the similarities and differences that exist between the documents, taking into account the time that the incidents occurred and the document source. This enables us to distinguish the document relations into *synchronic* and *diachronic* ones. In the synchronic level we try to identify the similarities and differences that exist between the various sources. In the diachronic level, on the other hand, we try to identify similarities and differences across time focusing on each source separately.

Another twofold distinction that we made through our study (Afantenos *et al.* 05b) concerns the type of evolution of an event, distinguishing between *linear* and *non-linear* evolution, and the rate of emission of the various news sources, distinguishing between *synchronous* and *asynchronous* emission of reports. Figure 1 depicts the major incidents for two different events: a linearly evolving event with synchronous emission and a non-linearly evolving one with asynchronous emission of reports. Whereas in the linearly evolving events the main incidents happen in constant and possibly predictable quanta of time,[1] in the non-linear events we can make no predictions as to when the next incident will occur. As you can see in Figure 1 we can have within a small amount of time an explosion of incidents followed by a long time of sparse incidents, etc.

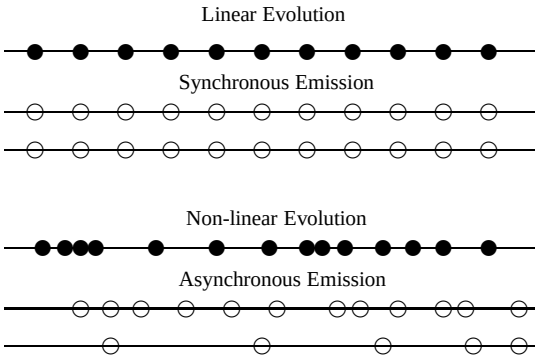

Figure 1: Linear and Non-linear evolution

In order to represent the various incidents that are described in each document, we introduce the notion of *messages*. Messages are composed from a name, which reflects the type of the incidents, and a list of arguments, which take their values from the domain ontology. Additionally, they

[1] This means that if the first news story q_0 comes at moment t_0, then we can assume that for each source the story q_n will come at time $t_n = t_0 + n * t$, where t is the constant amount of time that it takes for the news to appear.

have associated with them the *time* that the message refers to, as well as the document *source*.

The distinction between linear and non-linear evolution affects mainly the *synchronic relations*, which are used in order to identify the similarities and differences between two messages from different sources, at about the same time. In the case of linear evolution all the sources report in the same time. Thus, in most of the cases, the incidents described in each document refer to the time that the document was published. Yet, in some cases we might have temporal expressions in the text that modify the time that a message refers to. In such cases, before establishing a synchronic relation, we should associate this message with the appropriate time-tag. In the case of non-linear evolution, each source reports at irregular intervals, possibly mentioning incidents that happened long before the publication of the article, and which another source might have already mentioned in an article published earlier. In this case we shouldn't rely any more to the publication of an article, but instead rely on the *time* tag that the messages have (see section 2). Once this has been performed, we should then establish a *time window* in which we should consider the messages, and thus the relations, as synchronic.

In the following section, we make more concrete and formal the notion of the messages and relations. In section 3 we briefly present our methodology and describe its implementation through a particular case study. Section 4 presents in more detail the related work, and section 5 concludes presenting ongoing work on *non-linear* summarization and our future plans.

2 Some Definitions

In our approach (Afantenos *et al.* 05b; Afantenos *et al.* 04) the major building blocks for representing the knowledge on a specific event are: the *ontology* which encodes the basic entity types (concepts) and their instances; the *messages* for representing the various incidents inside the document; and the *relations* that connect those messages across the documents. More details are given below.

Ontology. For the purposes of our work, a domain ontology should be built. The ontology we use is a taxonomic one, incorporating *is-a* relations, which are later exploited by the messages and the relations.

Messages. In order to capture what is represented by several textual units, we introduce the notion of *messages*. A message is composed from four parts: its *type*, a list of *arguments* which take their values from the concepts of the domain *ontology*, the *time* that the message refers, and the *source* of the document that the message is contained. In other words, a message can be defined as follows:

$$\texttt{message\_type (arg}_1\texttt{, ... , arg}_n\texttt{)}$$
$$\text{where } \texttt{arg}_i \in \text{Domain Ontology}$$

Each message m is accompanied by the time ($m.\texttt{time}$) that it refers and its source ($m.\texttt{source}$). Concerning the source, this is inherited by the source of the document that contains the message. Concerning the time of the message, it is inherited by the publication time of the document, unless there exists a temporal expression in the text that modifies the time that a message refers. In this case, we should interpret the time-tag of the message, in relation to that temporal expression. A message definition may also be accompanied by a set of *constraints* on the values that the arguments can take. We would like also to note that messages are similar structures (although simpler ones) with the templates used in the MUC.[2] An example of a message definition will be given in the case study we present in section 3.

Relations. In order to define a relation in a domain we have to provide a *name* for it, and describe the conditions under which it will hold. The name of the relation is in fact *pragmatic* information, which we will be able to exploit later during the generation of the summary. The conditions that a relation holds are simply some rules which describe the *temporal distance* that two messages should have (0 for synchronic and more than 1 for diachronic) and the characteristics that the arguments of the messages should exhibit in order for the relation to hold.

Furthermore, it is crucial to note here the importance that time and source position have on the relations, apart from the values of the messages' arguments. Suppose, for example, that we have two identical messages. If they have the same temporal tag, but belong to different sources, then we have an *agreement* relation. If, on the other hand, they come from the same source but they have chronological distance one, then we speak of a *stability* relation. Finally, if they come from different sources and they have chronological distance more than two, then we have no relation at all. We also do not have a relation if the messages have different sources and different chronological distances. Thus we see that, apart from the characteristics that the arguments of a message pair should exhibit, the source and temporal distance also play a role for that pair to be characterized as a relation. In section 3 we will give concrete examples of messages and relations for a particular case study.

3 A Case Study of Linear Evolution

The methodology was originally presented in (Afantenos *et al.* 04). It involves four stages:

1. Corpus collection
2. Creation of a domain ontology
3. Specification of the messages
4. Specification of the relations

The topic we have chosen is that of the descriptions of football matches. In this domain, we have several events that evolve; for example, the performance of a player or

[2] `http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_toc.html`

a team as the championship progresses. According to the definitions we have given, the evolution of this domain is *linear*. The reason for this is that we have a match each week which is then being described by several sources.

As our methodology requires, in order to create multi-document summaries of evolving events, we have to provide some knowledge of the domain to the system. This knowledge is provided through the ontology and the specification of the messages and the relations, following the four steps described above.

3.1 Domain Knowledge

Corpus Collection. We manually collected descriptions of football matches, from various sources, for the period 2002-2003 of the Greek football championship. The language used in the documents was also Greek. This championship contained 30 rounds. We focused on the matches of a certain team, which were described by three sources. So, we had in total 90 documents.

Ontology Creation. An excerpt of the taxonomic ontology we have created is shown in Figure 2.

```
Degree                Card
Person                  Yellow
  Referee               Red
  Assistant Referee   Team
  Linesman            Temporal Concept
  Coach                 Minute
  Player                Duration
  Spectators              First Half
    Viewers               Second Half
    Organized Fans        Delays
Round                     Whole Match
```

Figure 2: An excerpt from the domain ontology

Messages' Specifications. We concentrated in the most important events, that is on events that evolve through time, or events that a user would be interested in knowing. At the end of this process we concluded on the following set of 23 message types:

Absent, Behavior, Block, Card, Change, Comeback, Conditions, Expectations, Final_Score, Foul, Goal_Cancelation, Hope_For, Injured, Opportunity_Lost, Penalty, Performance, Refereeship, Satisfaction, Scorer, Successive_Victories, Superior, System_Selection, Win

An example of full message specifications is shown in Figure 3. We should note that this particular message type is not accompanied by constraints. Also, associated with it we have the `time` and `source` tags.

performance (entity, in_what, time_span, value)

entity	:	player or team
in_what	:	Action Area
time_span	:	Minute or Duration
value	:	Degree

Figure 3: An example of message specifications

Specification of the Relations. We identified twelve cross-document relations, six on the synchronic and six on the diachronic axis (see Table 1).

Diachronic Relations	***Synchronic Relations***
– POSITIVE GRADUATION	– AGREEMENT
– NEGATIVE GRADUATION	– NEAR AGREEMENT
– STABILITY	– DISAGREEMENT
– REPETITION	– ELABORATION
– CONTINUATION	– GENERALIZATION
– GENERALIZATION	– PRECISENESS

Table 1: Synchronic and Diachronic Relations in the Football Domain

Since this was a pilot-study during which we examined mostly the feasibility of our methodology, we limited the study of the cross-document relations, in those ones that connect the *same* message types. Thus both the synchronic and the diachronic relations connect the same types, although further studies might reveal that different message types can be connected with some sort of relations. Furthermore, concerning the *diachronic* relations we limited our study in relations that have chronological distance only one.[3] Examples of such specifications for the message type `performance` are shown in Figure 4.

Performance
Assuming we have the following two messages:

performance_1 (entity_1, in_what_1, time_span_1, value_1)
performance_2 (entity_2, in_what_2, time_span_2, value_2)

Then we have a Diachronic relation if

(performance_1`.time` $<$ performance_2`.time`) and
(performance_1`.source` = performance_2`.source`)

and a Synchronic relation if

(performance_1`.time` = performance_2`.time`) and
(performance_1`.source` $\neq$ performance_2`.source`)

More specifically, we have the following Synchronic and Diachronic relations:

Diachronic Relations

- **Positive Graduation** iff
  (entity_1 = entity_2) and (in_what_1 = in_what_2) and (time_span_1 = time_span_2) and ($\text{value}_1 < \text{value}_2$)
- **Stability** iff
  (entity_1 = entity_2) and (in_what_1 = in_what_2) and (time_span_1 = time_span_2) and (value_1 = value_2)
- **Negative Graduation** iff
  (entity_1 = entity_2) and (in_what_1 = in_what_2) and (time_span_1 = time_span_2) and ($\text{value}_1 > \text{value}_2$)

Synchronic Relations

- **Agreement** iff
  (entity_1 = entity_2) and (in_what_1 = in_what_2) and (time_span_1 = time_span_2) and (value_1 = value_2)
- **Near Agreement** iff
  (entity_1 = entity_2) and (in_what_1 = in_what_2) and (time_span_1 = time_span_2) and ($\text{value}_1 \approx \text{value}_2$)
- **Disagreement** iff
  (entity_1 = entity_2) and (in_what_1 = in_what_2) and (time_span_1 = time_span_2) and ($\text{value}_1 \neq \text{value}_2$)

Figure 4: Specifications of Relations

[3] Chronological distance zero makes the relations synchronic.

A question that can arise is the following: *How does time affect the relations you create?* To answer that question, imagine having two identical messages, in different documents. If the documents have chronological distance zero, then we have an *agreement* relation. If the messages come from the same source but have chronological distance 1, then we have a *stability* relation. Finally, if the messages come from different sources and have chronological distance more than one, then we have no relation at all. Thus, indeed, time does affect the relations.

An Example At this point we would like to give a more concrete example. Two sources, A and B, for a particular match, describe the performance of a player as follows:

A The performance of Nalitzis, for the whole match was mediocre.

B In general, we can say that Nalitzis performed modestly, throughout the match.

The messages that represent those two sentences are the following:

A `performance (Nalitzis, general, whole_match, 50)`

B `performance (Nalitzis, general, whole_match, 50)`

The number 50 represents the mediocre performance of the player, since the `degree` is realized as an integer in the scale of 0 to 100. According to the specifications of the relations (see Figure 4) we would have an *Agreement* synchronic relations between those two messages. In the next game we have the following description:

A Nalitzis shown an excellent performance throughout the game.

The message that results from this sentence is the following:

A `performance (Nalitzis, general, whole_match, 100)`

Now, between the two messages from source A we have a *Positive Graduation* diachronic relation.

3.2 The System

Our summarization system is a query-based one, since the summary is an answer to a natural language query that a user has posed. Such queries concern the evolution of several events in the domain. In order to create the summaries we have to extract, from the documents, the messages with their arguments, and the relations that connect them, and subsequently organize them into a structure which we call a *grid* (see Figure 5). This grid reflects exactly the fact that the domain that we have used in this case study exhibits linear evolution. If we take a horizontal "slice" of the grid, then we will have descriptions of events from all the sources, for a particular time unit. If, on the other hand, we take a vertical "slice" of the grid, then we have the description of the evolution of an event from a particular source.

In order to extract the messages from the documents, our system employs an Information Extraction (IE) subcomponent. Relations between the messages are identified according to the conditions associated with each one. After the user has issued the query, the system identifies the various messages that are relevant to this query, as well as the relations that connect them. Thus, in essence the system *extracts a subgrid* from the original grid which is, in fact, the answer to the user query. This subgrid is passed to a Natural Language Generation (NLG) subcomponent which creates the final summary.

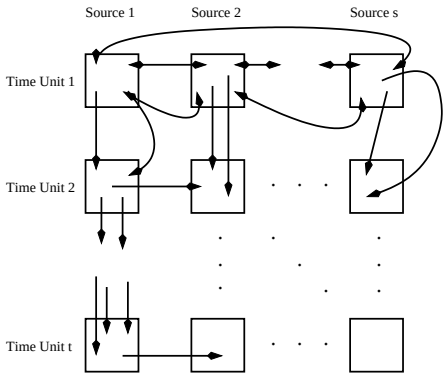

Figure 5: The Grid structure with Synchronic and Diachronic Relations

3.2.1 Messages Extraction

This subsystem was developed using the *Ellogon* text engineering platform.[4] Its architecture is depicted in Figure 6. It involves the following processing stages.

Preprocessing. This stage includes the tokenization, sentence splitting and the Named Entity Recognition and Classification (NERC) sub-stages. During NERC, we try to identify the Named Entities (NEs) in the documents and classify them into the categories that the ontology proposes.

The next two processing stages are the core of message extraction. In the first one we try to identify the *type* of each extracted message, while in the second we try to fill its *argument values*.

Message Classification. Concerning the identification of the message types, we approached it as a classification problem. From a study that we carried out, we concluded that in most of the cases the mapping from sentences to messages was one-to-one, *i.e.* in most of the cases one sentence corresponded to one message. Of course, there were cases in which one message was spanning more than one sentence, or that one sentence was containing more than one message. We managed to deal with such cases during the arguments' filling stage.

In order to perform our experiments we used a bag-of-words approach according to which we represented each sentence as a vector from which the stop-words and the words with low frequencies (four or less) were removed. The features used are divided into two categories: *lexical* and *semantic*. As lexical features we used the words of the

[4] `www.ellogon.org`

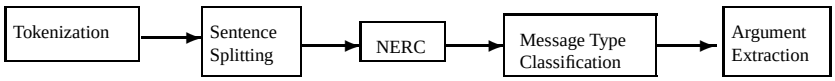

Figure 6: The message extraction subsystem

sentences both stemmed and unstemmed. As semantic features we used the *NE types* that appear in the sentence. Of course, in order to perform the training phase of the experiments, in each of the vectors we appended the *class* of the sentence, *i.e.* the type of message; in case a sentence did not corresponded to a message we labeled that vector as belonging to the class *None*. This resulted into *four* series of vectors and corresponding experiments that we performed.

In order to perform the classification experiments we used the WEKA platform (Witten & Frank 00). The Machine Learning algorithms that we used where three: *Naïve Bayes*, *LogitBoost* and *SMO*. For the last two algorithms, apart from the default configuration, we performed some more experiments concerning several of their arguments. Thus for the LogiBoost we experimented with the number of iterations that the algorithm performs and for the SMO we experimented with the complexity constant, with the exponent for the polynomial kernel and with the gamma for the RBF kernel. For each of the above combinations we performed a *ten-fold cross-validation* with the annotated corpora that we had. The results of the above experiments are presented in Table 2.

Taking a look at that table there are several remarks that we can make. Firstly, the LogitBoost and the SMO classifiers that we used outperformed, in all the cases, the Naïve Bayes which was our baseline classifier. Secondly, the inclusion of the NE types in the vectors gave a considerable enhancement to the performance of all the classifiers. This is rather logical, since almost all the messages contain in their arguments NEs. The third remark, concerns the stemmed and the unstemmed results. As we can see from the table, the algorithms that used vectors which contained unstemmed words outperformed the corresponding algorithms which used vectors whose words had been stemmed. This is rather counterintuitive, since in most of the cases using stemming one has better results.

Ultimately, the algorithm that gave the best results, in the experiments we performed, was the SMO with the default configuration for the unstemmed vectors which included information on the NE types. This classifier managed to correctly classify 2974 out of 3735 messages (including the *None* class) or about 80% of the messages. Thus, we integrated this trained classifier in the message extraction subsystem, which you can see in Figure 6.

Arguments' Filling In order to perform this stage we employed several domain-specific heuristics. Those heuristics take into account the constraints of the messages, if they do have. As we noted above, one of the drawbacks of our classification approach is that there are some cases in which we do not have a one-to-one mapping from sentences to messages. During this stage of message extraction we used heuristics to handle many of these cases.

In Table 3 we show the final performance of the subsystem as a whole, when compared against manually annotated messages on the corpora used. Those measures concern only the message types. As you can see from that table although the vast majority of the messages extracted are correct, these represent 68% of all the messages.

Precision	:	91.1178
Recall	:	67.7810
F-Measure	:	77.7357

Table 3: Evaluation of the messages' extraction stage

3.2.2 Extraction of Relations

As is evident from Figure 4, once we have identified the messages in each document and we have placed them in the appropriate position in the grid, then it is fairly straightforward, through their specifications, to identify the cross-document relations among the messages.

In order to achieve that, we implemented a system which was written in Java. This system takes as input the extracted messages with their arguments from the previous subsystem and it is responsible for the incorporation of the ontology, the representation of the messages and the extraction of the synchronic and diachronic cross-document relations. Ultimately, through this system we manage to represent the *grid*, which carries an essential role for our summarization approach.

The reason for this is that since our approach is a query based one, we would like to be able to pose queries and get the answers from the grid. The system that we have created implements the API through which one can pose queries to the grid, as well as the mechanism that extracts from the whole grid structure the appropriate messages and the relations that accompany them, which form an answer to the question. Those extracted messages and relations form a sub-grid which can then be passed to an NLG system for the final creation of the summary.

Concerning the statistics of the extracted relation, these are presented in Table 4. The fact that we have lower statistical measures on the relations, in comparison with the message types, can be attributed to the argument extraction subsystem, which does not perform as well as the message classification subsystem.

Precision	:	89.0521
Recall	:	39.1789
F-Measure	:	54.4168

Table 4: Recall, Precision and F-Measure on the relations

Classifier		Correctly Classified Instances	Classifier		Correctly Classified Instances
Without NE types			Including NE types		
stemmed	Naïve Bayes	60.6693 %	stemmed	Naïve Bayes	63.8286 %
	LogitBoost default	72.7443 %		LogitBoost default	78.0991 %
	LogitBoost $I = 5$	71.8876 %		LogitBoost $I = 5$	76.1981 %
	LogitBoost $I = 15$	72.2892 %		LogitBoost $I = 15$	78.2062 %
	SMO default	73.6011 %		SMO default	75.9839 %
	SMO $C = 0.5$ $E = 0.5$ $G = 0.001$	68.9692 %		SMO $C = 0.5$ $E = 0.5$ $G = 0.001$	72.5301 %
	SMO $C = 1.5$ $E = 1.5$ $G = 0.1$	74.4578 %		SMO $C = 1.5$ $E = 1.5$ $G = 0.1$	75.7965 %
unstemmed	Naïve Bayes	62.2758 %	unstemmed	Naïve Bayes	64.2035 %
	LogitBoost default	75.8768 %		LogitBoost default	78.9023 %
	LogitBoost $I = 5$	74.9398 %		LogitBoost $I = 5$	77.4565 %
	LogitBoost $I = 15$	76.6533 %		LogitBoost $I = 15$	79.4645 %
	SMO default	79.2503 %		SMO default	79.6252 %
	SMO $C = 0.5$ $E = 0.5$ $G = 0.001$	75.2343 %		SMO $C = 0.5$ $E = 0.5$ $G = 0.001$	76.8675 %
	SMO $C = 1.5$ $E = 1.5$ $G = 0.1$	77.9920 %		SMO $C = 1.5$ $E = 1.5$ $G = 0.1$	78.5007 %

Table 2: The results from the classification experiments

As of writing this paper, everything has been implemented except the mechanism that transforms the natural language queries to the API that will extract the subgrid. Additionally, we do not have a connection with an NLG system, but instead we have implemented some simple template-based mechanism.

4 Related Work

The work that we present in this paper is concerned with multi-document summarization of events that evolve through time. Of course, we are not the first to incorporate directly, or indirectly, the notion of time in our approaches to summarization. (Lehnert 81), for example, attempts to provide a theory for what she calls *narrative summarization*. Her approach is based on the notion of "plot units", which connect *mental states* with several relations, and are combined into very complex patterns. This approach is a single-document one and was not implemented. Recently, (Mani 04) attempts to revive this theory of narrative summarization, although he also does not provide any concrete computational approach for its implementation.

From a different viewpoint, (Allan *et al.* 01) attempt what they call *temporal summarization*. In order to achieve that, they take the results from a Topic Detection and Tracking system for an event, and they put all the sentences one after the other in a chronological order, regardless of the document that it belonged, creating a stream of sentences. Then they apply two statistical measures *usefulness* and *novelty* to each ordered sentence. The aim is to extract those sentences which have a score over a certain threshold. This approach does not take into account the document sources, and it is not concerned with the evolution of the events; instead they try to capture novel information.

As we have said, our work requires some domain knowledge which is expressed through the ontology, and the messages' and relations' specifications. A system which is based also on domain knowledge is SUMMONS (Radev & McKeown 98; Radev 99). The domain knowledge for this system comes from the specifications of the MUC conferences. This system takes as input several MUC templates and, applying a series of operators, it tries to create a baseline summary, which is then enhanced by various named entity descriptions collected from the Internet. One can argue that the operators that SUMMONS uses resemble our cross-document relations. This is a superficial resemblance, since our relations are divided into *synchronic* and *diachronic*, thus reporting similarities and differences in two different directions: source and time. Additionally our system is a query-based one.

Concerning now the use of relations, (Salton *et al.* 97) for example, try to extract paragraphs from a single document by representing them as vectors and assigning a relation between the vectors if their similarity exceeds a certain threshold. Then, they present various heuristics for the extraction of the best paragraphs.

Finally, (Radev 00) proposed the Cross-document Structure Theory (CST) which incorporated a set of 24 domain independent relations that exist between various textual units across documents. In a later paper (Zhang *et al.* 02) reduce that set into 17 relations and perform some experiments with human judges. Those experiments reveal several interesting results. For example, human judges annotate only sentences, ignoring the other textual units (phrases, paragraphs, documents) that the theory suggests. Additionally, we see a rather small inter-judge agreement concerning the type of relation that connects two sentences. (Zhang *et al.* 03) and (Zhang & Radev 04) continue the work with some experiments, during which they use Machine Learning techniques to identify the cross-document relations. We have to note here that although a general *pool* of cross-document relations might exist, we believe, in contrast with (Radev 00), that those relations are dependent on the domain, in the sense that one can choose from this pool the appropriate subset of relations for the domain under consideration, possibly enhancing this subset with completely domain specific relations that will suit ones needs. Another significant difference from our work, is that our main goal is to create summaries that show the evolution of an event, as well as the similarities or differences that the sources have during the evolution of an event.

5 Conclusions and Future Work

The aim of this paper was to present our approach to the problem of *multi-document summarization of evolving events*. We divide the evolution of the events into linear and non-linear. In order to tackle the problem, we

introduced cross-document relations which represent the evolution of the events in two axes: *synchronic* and *diachronic*. Those relations connect messages, which represent the main events of the domain, and are dependent on the domain ontology. We also presented, through a case study, an implementation for a linearly evolving domain, namely that of the descriptions of football matches. The system we have built automatically extracts the messages and the synchronic and diachronic relations from the text. A particular point of concern is the recall (approximately 40%) of the relations' extraction sub-system, which is due to the heuristics used for the filling the arguments of the messages. Apart from enhancing our heuristics, we also plan to study their effect on the quality of the generated summary.

Currently we are working on a more complicated domain, namely that of events with hostages, whose evolution, according to the specification that we gave in the introduction of this paper, can be characterized as non-linear. The main challenges in non-linear evolution concern the synchronic relations. A related problem, which we investigate, is that of the *temporal expressions* which may make several messages refer back in time, in relation to the publication time of the article that contains the messages. We also examine in depth the role that time has on the relations. Additionally, we examine the existence of relations between different message types. Concerning now the classification experiments and the argument extraction, we intend to enhance them by adding more semantic features incorporating also the Greek WordNet.[5]

References

(Afantenos *et al.* 04) Stergos D. Afantenos, Irene Doura, Eleni Kapellou, and Vangelis Karkaletsis. Exploiting cross-document relations for multi-document evolving summarization. In G. A. Vouros and T. Panayiotopoulos, editors, *Methods and Applications of Artificial Intelligence: Third Hellenic Conference on AI, SETN 2004*, volume 3025 of *Lecture Notes in Computer Science*, pages 410–419, Samos, Greece, May 2004. Springer-Verlag Heidelberg.

(Afantenos *et al.* 05a) Stergos D. Afantenos, Vangelis Karkaletsis, and Panagiotis Stamatopoulos. Summarization from medical documents: A survey. *Journal of Artificial Intelligence in Medicine*, 33(2):157–177, February 2005.

(Afantenos *et al.* 05b) Stergos D. Afantenos, Konstantina Liontou, Maria Salapata, and Vangelis Karkaletsis. An introduction to the summarization of evolving events: Linear and non-linear evolution. In *Natural Language Understanding and Cognitive Science NLUCS - 2005*, pages 91–99, Maiami, USA, May 2005.

(Allan *et al.* 98) James Allan, Jaime Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. Topic detection and tracking pilot study: Final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218, February 1998.

(Allan *et al.* 01) James Allan, Rahuk Gupta, and Vikas Khandelwal. Temporal summaries of news stories. In *Proceedings of the ACM SIGIR 2001 Conference*, pages 10–18, 2001.

(Lehnert 81) Wendy G. Lehnert. Plot units: A narrative summarization strategy. In W. G. Lehnert and M. H. Ringle, editors, *Strategies for Natural Language Processing*, pages 223–244. Erlbaum, Hillsdale, New Jersey, 1981.

(Mani & Bloedorn 99) Inderjeet Mani and Eric Bloedorn. Summarizing similarities and differences among related documents. *Information Retrieval*, 1(1):1–23, 1999.

(Mani 01) Inderjeet Mani. *Automatic Summarization*, volume 3 of *Natural Language Processing*. John Benjamins Publishing Company, Amsterdam/Philadelphia, 2001.

(Mani 04) Inderjeet Mani. Narrative summarization. *Journal Traitement Automatique des Langues (TAL): Special issue on "Le résumé automatique de texte: solutions et perspectives"*, 45(1), Fall 2004.

(Radev & McKeown 98) Dragomir R. Radev and Kathleen R. McKeown. Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3):469–500, September 1998.

(Radev 99) Dragomir R. Radev. *Generating Natural Language Summaries from Multiple On-Line Sources: Language Reuse and Regeneration*. Unpublished PhD thesis, Columbia University, 1999.

(Radev 00) Dragomir R. Radev. A common theory of information fusion from multiple text sources, step one: Cross-document structure. In *Proceedings of the 1st ACL SIGDIAL Workshop on Discourse and Dialogue*, Hong Kong, October 2000.

(Salton *et al.* 97) Gerald Salton, Amit Singhal, Mandar Mitra, and Chris Buckley. Automatic text structuring and summarization. *Information Processing and Management*, 33(2):193–207, 1997.

(Witten & Frank 00) Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco, 2000.

(Zhang & Radev 04) Zhu Zhang and Dragomir Radev. Learning cross-document structural relationships using both labeled and unlabeled data. In *Proceedings of IJC-NLP 2004*, Hainan Island, China, March 2004.

(Zhang *et al.* 02) Zhu Zhang, Sasha Blair-Goldensohn, and Dragomir Radev. Towards cst-enhanced summarization. In *Proceedings of AAAI-2002*, August 2002.

(Zhang *et al.* 03) Zhu Zhang, Jahna Otterbacher, and Dragomir Radev. Learning cross-document structural relationships using boosting. In *Proccedings of the Twelfth International Conference on Information and Knowledge Management CIKM 2003*, pages 124–130, New Orleans, Louisiana, USA, November 2003.

[5] `www.ceid.upatras.gr/Balkanet/resources.htm`