Summarizing Reports on Evolving Events Part II: Non-Linear Evolution

Stergos D. Afantenos

LINA, (UMR CNRS 6241), Université de Nantes, France

Abstract. News sources, when covering an event, are emitting various reports during the course of the event's evolution. This paper focuses on the task of summarizing such reports. After discussing the nature of evolving events (dividing them into linearly and non-linearly evolving events), we present a methodology for the creation of summaries of evolving events as they are being described by multiple sources. At the core of this methodology lies the notion of Synchronic and Diachronic Relations (SDRs) whose aim is the identification of similarities and differences across documents. SDRs do not connect textual elements inside the text but some structures which we call *messages*. We present the application of our methodology via a case study.

1 Introduction

Using a news aggregation service, such as Google News for example, one realizes that most events being described contain hundreds or even thousands of reports. This makes it humanly impossible to keep track of the evolution of such events by reading all the articles. A possible solution to this problem could be the task of *automatic text summarization*. In this paper we present a methodology whose aim is the creation of summaries from multiple reports, emitted by various sources, on the same event. After presenting, in section 2, a distinction between linearly and non-linearly evolving events, in section 3 we present our methodology. In section 4 we apply this methodology in a particular case study of events that evolve non-linearly. We present our future work and conclude with section 5.

2 Evolving Events: What are They and How do They Evolve?

As we have said, this paper is on the summarization of events that evolve through time. Two questions that could naturally arise at this point are: (a) what is an event and, (b) how do events evolve?

The first question is actually a question to which researchers on the DARPA initiative of Topic Detection and Tracking (TDT) have extensively pondered throughout their research. In the first TDT, a distinction was made between the notions of *topic*, *event* and *activity*. An event was defined as "something that happens at some specific time and place" [7, 10], while a topic was considered as being the general class of "similar" events (*e.g.* “volcano explosions” is a topic, while the explosion of Krakatoa on August 26 1883 was a particular event). Activities have been defined as “a connected set of actions that have a common focus or purpose” [10, p 3]. According always to those first definitions, each event contained several activities. Although the above definitions seem to be quite clear, some further reflection reveals that the distinction between a topic and an event is not always clear. Events, for example, could span more than a specific point in time, having thus a duration measured in hours, days or even longer time spans. Problems of this nature, among other considerations, have driven the researchers involved in later TDT conferences to change their perspectives on what the notions of topic and events mean. According then to later consensus “a topic is defined to be a set of news stories that are strongly related by some seminal real-world event” [6, p 2]. In other words, according to this new viewpoint a topic is nothing else but a collection of strongly related news reports which have been triggered by an initial event, and which follow the evolution of this event.

The above distinction between topics and events is a point of view that we share with the community of TDT and one which we will adopt in the context of this paper. Let us now try to answer the second question posed at the beginning of this section: how do topics evolve? This question can be more precisely formulated as “how do the events that constitute a topic evolve?” Concerning this question we distinguish between two types of evolving events: *linearly* and *non-linearly* evolving events. In the case of a topic which exhibits a linear evolution of events, the events are happening in predictable and possibly constant quanta of time. In the case of topics which exhibit non-linear evolution of events, in contrast, we cannot distinguish any such regularity. Topics with linearly evolving events have a fair proportion in the world. They can range from descriptions of various athletic incidents to quarterly reports that an organization is publishing. On the other hand, one can argue that most of the topics that we find in the news stories contain non-linearly evolving events. They can vary from political ones to airplane crashes or terrorist incidents. An auxiliary question that we could pose concerns the rate with which the various sources emit their reports. In this context we can distinguish between *synchronous* and *asynchronous* emission of reports. In the first case, the sources publish almost simultaneously their reports, whilst in the second case each source follows its own agenda in publishing their reports. In most of the cases, when we have a topic with linearly evolving events we will also have a synchronous emission of reports, since the various sources can easily adjust to the pattern of the evolution of the topic. This cannot be said for the case of non-linear evolution, resulting thus in asynchronous emission of reports by the various sources.

In this paper we will present a methodology for the creation of summaries from evolving events, with a particular case-study to a topic which exhibits non-linear evolution of events (section 4).

3 Methodology

Our methodology is divided into two major phases, the *topic analysis phase* and the *implementation phase*. The goal of the first phase is the definition of the “building blocks” with which we are going to represent the knowledge for the general class of topics that we need to create summaries for. Those “building blocks” are: the *ontology* which encodes the basic entity types; the *messages* for representing the various actions inside the document; and the *relations* that synchronically and diachronically connect those messages across the documents. Once the above building blocks have been defined, we pass then to the implementation phase whose aim is the identification and classification of the instances of the ontology entities, the extraction of the messages and finally the connection of the messages with the SDRs. The result will be the creation of an abstract representation of the information contained in the initial set of documents, in a graph whose nodes are the messages and whose vertices are the SDRs. We call this graph the *grid*. The grid will later be passed over to a Natural Language Generation (NLG) system in order to create the final summary. In this section we will provide more information on the steps involved during the topic analysis phase. The implementation phase will be the focus of section 4.

3.1 Ontology

According to the consensus that has been reached in the area of ontology creation [8, 9, 11], there are four steps involved during this process: *specification*, *conceptualization*, *formalization* and *implementation*.[1] For the purposes of our study we have followed the above four steps in order to create our ontology.

3.2 Messages

Messages are semantic units of information which are meant to represent the actions that are happening inside a given topic, connecting them with the entities involved with those actions. A message thus is composed of two parts: its *name* and *a list of arguments* which represent the ontology concepts involved in the action that the message represents. In addition, each message is accompanied by information on the source from which it was emitted and its publication time, as well as by information on the time at which it refers. Usually the publication and referring time will be equal unless some temporal expressions are found in the text that alter the time to which the message refers. Thus, a message can be defined as follows.

$$m = \texttt{message\_type (arg}_1\texttt{, ... , arg}_n\texttt{)}$$

where $\texttt{arg}_i \in$ Topic Ontology, $i \in \{1, \ldots, n\}$, and:

$|m|_{\texttt{source}}$: the source which contained the message,
$|m|_{\texttt{pub\_time}}$: the publication time of the message,
$|m|_{\texttt{ref\_time}}$: the referring time of the message.

The aim of the topic analysis phase is the creation of a list with all the messages that can be found in a given topic. The way that this is currently performed is by studying a given corpus and abstracting off the message types as well as the entities involved in those message types. Later, during the implementation phase, the actual instances of

[1] Actually, a fifth step of *maintenance* exists as well. At the current state of our research, this step is not included.

the messages have to be identified in the corpora. We have to note that the above process is similar to the Information Extraction paradigm, where a given set of templates, representative to a given domain, is provided to a system, which has later to fill in the slots of the templates. The method that we use in order to identify the instances of the messages and fill in their arguments is presented in section 4.

3.3 Synchronic and Diachronic Relations

An essential task for the process of Multi-Document Summarization—as well as for several other Natural Language Processing tasks—is the identification of the similarities and differences that exist between various sources. When it comes to the task of creating summaries from evolving events, we believe that this task entails the description of the event's evolution, as well as the designation of the points of conflict or agreement between the sources, as the event evolves. In order to capture the evolution of an event as well as the conflict, agreement or variation between the sources, we introduce the notion of *Synchronic and Diachronic Relations (SDRs)*. Synchronic relations try to identify the degree of agreement, disagreement or variation between the various sources, at about the same time frame. Diachronic relations, on the other hand, try to capture the evolution of an event as it is being described by one source. SDRs hold between two messages, and a definition of an SDR consists of the following four fields:

1. The relation's type (*i.e.* Synchronic or Diachronic).
2. The relation's name.
3. The set of pairs of message types that are involved in the relation.
4. The constraints that the corresponding arguments of each of the pairs of message types should have.

The name of the relation carries *semantic* information which, along with the messages that are connected with the relation, are later being exploited by the NLG component in order to produce the final summary. The aim of the Synchronic relations is to capture the degree of agreement, disagreement or variation that the various sources have for the *same time-frame*. The same time-frame is determined by the messages' referring time. The aim of Diachronic relations, on the other hand, is to capture the evolution of an event as it is being described by *one source*. Thus, all messages that belong to the same source and have a different referring time are initially considered as candidates for connection with a Diachronic Relation. SDRs could hold either between messages of the same type or messages of different types. Examples of SDRs will be provided in section 4.

4 A Case-Study

As we have said in section 2, the evolution of topics can be either linear or non-linear. The methodology that we have presented in section 3 applies for both kinds of evolution. In previous studies [1–5] we have applied our methodology in a topic which exhibits linear evolution. In this paper we would like to present a case study of a class of topics which exhibit non-linear evolution. The class of topics that we have chosen to work with are the terroristic incidents which involve hostages.

4.1 Topic Analysis

Corpus Collection. A prerequisite step before moving to the stages involved in the topic analysis phase is the collection of a corpus. The perusal of this corpus will not only be limited for the definition of the ontology, messages and SDRs, but it will also be used for annotation purposes, something which will be useful during the training phase of the various Machine Learning (ML) algorithms involved in the implementation phase.

The corpora that we collected come from five different topics: the hijacking of an airplane from the Afghan Airlines in February 2000, a Greek bus hijacking from Albanians in July 1999, the kidnapping of two Italian reporters in Iraq in September 2004, the kidnapping of a Japanese group in Iraq in April 2004, and finally the hostages incident in the Moscow theater by a Chechen group in October 2004. In total we collected and examined 163 articles from 6 sources.

Figure 1 (left part) presents the statistics, concerning the number of documents and words contained therein, for each event separately.

Event	Documents	Words
Airplane Hijacking	33	7008
Bus Hijacking	11	12416
Italians Kidnaping	52	21200
Japanese Kidnaping	18	10075
Moscow Theater	49	21189

```
Person                        Place
  Offender                      Country
  Hostage                       City
  Demonstrators               Vehicle
  Rescue Team                   Bus
  Relatives                     Plane
  Professional                  Car
  Governmental Executive
```

Fig. 1. Statistics on each topic (left) and an excerpt from the topic ontology (right).

Ontology Creation. For the creation of the ontology we followed the guidelines presented in section 3.1. An excerpt of the final ontology can be seen in Figure 1 (right part).

Messages' Specifications. The specification of the messages involves the specification of the message types as well as the ontology entities that are involved in each of the messages. After studying the corpora we concluded in the 48 messages shown in the left part of figure 2. Full specifications for two particular messages can be seen in the right part of the same figure.

Relations' Specifications. After studying the corpora we concluded in 15 Synchronic and Diachronic Relations (SDRs) as shown in the left part of figure 3. Examples of actual relations' specifications can be seen in the right part of the same figure.

4.2 Implementation

The stages involved during the implementation phase can be seen in figure 4. The result of the processing will be the creation of a structure that we call the *grid*. The grid is a graph whose nodes are the messages and whose vertices are the SDRs. The grid is later passed over an NLG system in order to create the final summary. In the following we will present in detail each of the stages involved in the summarization process.

`free`	`ask_for`	`located`
`kill`	`aim_at`	`inform`
`hold`	`kidnap`	`organize`
`deny`	`arrive`	`announce`
`enter`	`arrest`	`transport`
`help`	`armed`	`negotiate`
`meet`	`leave`	`threaten`
`start`	`end`	`work_for`
`put`	`return`	`hijack`
`lead`	`accept`	`trade`

negotiate (who, with_whom, about)
who : Person
whom : Person
about : Activity

free (who, whom, from)
who : Person
whom : Person
from : Place $\vee$ Vehicle

Fig. 2. Excerpt from the message types (left) and example of message specifications (right).

Synchronic Relations (same message types)
AGREEMENT
ELABORATION
DISAGREEMENT
SPECIFICATION

Diachronic Relations (same message types)
REPETITION
CHANGE OF PERSPECTIVE
CONTINUATION
IMPROVEMENT
DEGRADATION

Diachronic Relations (different message types)
CAUSE
FULFILLMENT
JUSTIFICATION
CONTRIBUTION
CONFIRMATION
MOTIVATION

In the following we will assume that we have the following messages

```
negotiate (who_a, with_whom_a, about_a)
free (who_b, whom_b, from_b)
free (who_c, whom_c, from_c)
```

The specifications for the relations are the following:

Relation Name:	*AGREEMENT*	*IMPROVEMENT*
Relation Type:	Synchronic	Diachronic
Pairs of Messages:	{<`free, free`>}	{<`negotiate, free`>}
Constraints on the arguments:	($\texttt{who}_b$ = $\texttt{who}_c$) $\wedge$ ($\texttt{whom}_b$ = $\texttt{whom}_c$) $\wedge$ ($\texttt{from}_b$ = $\texttt{from}_c$) $\wedge$	($\texttt{who}_a$ = $\texttt{who}_b$) $\wedge$ ($\texttt{about}_a$ = `free`)

Additionally, the messages should satisfy as well the constraints on the source and referring time in order to be candidates for a Synchronic or Diachronic Relation. In other words, the messages m_1 and m_2 will be candidates for a Synchronic Relation if

$$|m_1|\texttt{source} \neq |m_2|\texttt{source}$$
$$|m_1|\texttt{ref\_time} = |m_2|\texttt{ref\_time}$$

and candidates for a Diachronic Relation if

$$|m_1|\texttt{source} = |m_2|\texttt{source}$$
$$|m_1|\texttt{ref\_time} > |m_2|\texttt{ref\_time}$$

Fig. 3. Synchronic and Diachronic Relations (left) and example of their specifications (right).

Preprocessing. The preprocessing that we performed in our corpora involved the processes of tokenization, sentence splitting and part of speech tagging.

Entities Recognition and Classification. The aim of this stage is the identification of the various textual elements in the input documents that represent an ontology concept, and their classification into the appropriate ontology concept. Take for example the word "passenger". Depending on the context, this textual element could be either an instance of a `Hostage`, an instance of an `Offender`, or nothing at all (see again the right part of figure 1 for an excerpt of the ontology).

The approach that we used in order to attack this problem involves the use of ML techniques. More specifically we opted in using a *cascade of classifiers* which consists of three levels. The first level of the cascade is a binary classifier which determines whether a textual element in the input text is an instance of an ontology concept or not.

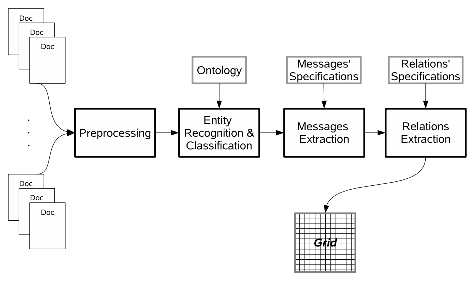

Fig. 4. The summarization system.

At the second level, the classifier takes the instances of the ontology concepts of the previous level and classifies them under the top-level ontology concepts. Finally at the third level we had a specific classifier for each top-level ontology concept, which classifies the instances in their appropriate sub-concepts. For all the levels of this cascade of classifiers we used the WEKA platform. More specifically we used three classifiers: *Naïve Bayes*, *LogitBoost* and *SMO*, varying the input parameters of each classifier. We will analyze each level of the cascade separately. For the first level of the cascade, we experimented with using from one to up to five context tokens around a candidate ontology concept. The features that we used included the token types, the part-of-speech types, as well as their combination. After performing a tenfold cross-validation using the annotated corpora, we found that the classifier which yielded the best results was LogitBoost with 150 boost iterations, using only the token types and a context window of four tokens. For the second level, we created a series of experiments which took into consideration one to up to five tokens before and after the textual elements, as well as the tokens which comprised the textual element itself. The features that we used were the token types, the part-of-speech types, and their combination. The classifier that yielded the best results, after performing a tenfold cross-validation, was LogitBoost with 100 boost iterations with a context of size one, and using as features the token types and part-of-speech types for each token. The final level of the cascade of classifiers consists of a specialized classifier for each top-level ontology concept. In this series of experiments we took as input only the nouns that were contained in each textual element, discarding all the other tokens. The combined results from the cascade of classifiers, after performing a tenfold cross-validation, are shown in Table 1. The last column in that table, represents the classifier used in the third level of the cascade. The parameter *I* in the LogitBoost classifier represents the boost cycles. For conciseness we present only the evaluation results for each top-level ontology concept.

Table 1. The combined results of the cascade of classifiers.

Class	Precision	Recall	F-Measure	Classifier
Person	75.63%	83.41%	79.33%	SMO
Place	64.45%	73.03%	68.48%	LogitBoost (I=700)
Activity	76.86%	71.80%	74.25%	LogitBoost (I=150)
Vehicle	55.00%	45.69%	49.92%	Naïve Bayes
Media	63.71%	43.66%	51.82%	LogitBoost (I=150)

Messages Extraction. This stage consists of three sub-stages. At the first one we try to identify the message types that exist in the input documents, while at the second we try to fill in the messages' arguments with the instances of the ontology concepts identified in the previous stage. The third sub-stage includes the identification of the temporal expressions that might exist in the text, and the normalization of the messages' referring time, in relation to the document's publication time.

In most of the cases we had a one-to-one mapping between message types and sentences. We used once again an ML approach, using *lexical* and *semantic* features for the creation of the vectors. As lexical features we used a fixed number of verbs and nouns occurring in the sentences. Concerning the semantic features, we used two kinds of information. The first one was a numerical value representing the number of the top-level ontology concepts that were found in the sentences. Thus the created vectors had eight numerical slots, each one representing one of the top-level ontology concepts. Concerning the second semantic feature, we used what we have called *trigger words*, which are several lists of words, each one "triggering" a particular message type. Thus, we allocated six slots—the maximum number of trigger words found in a sentence—each one of which represented the message type that was triggered, if any. In order to perform our experiments, we used the WEKA platform. The algorithms that we used were again the Naïve Bayes, LogitBoost and SMO, varying their parameters during the series of experiments that we performed. The best results were achieved with the LogitBoost algorithm, using 400 boost cycles. More specifically the number of correctly classified message types were 78.22%, after performing a ten-fold cross-validation on the input vectors.

The second sub-stage is the filling in of the messages' arguments. In order to perform this stage we employed several domain-specific heuristics which take into account the results from the previous stages. It is important to note here that although we have a one-to-one mapping from sentences to message types, it does not necessarily mean that the arguments (*i.e.* the extracted instances of ontology concepts) of the messages will also be in the same sentence. There may be cases where the arguments are found in neighboring sentences. For that reason, our heuristics use a window of two sentences, before and after the one under consideration, in which to search for the arguments of the messages, if they are not found in the original one. The total evaluation results from the combination of the two sub-stages of the messages extraction stage are shown in table 2. As in the previous cases, we also used a tenfold cross-validation process for the evaluation of the ML algorithms.

The last of the three sub-stages, in the messages extraction stage, is the identification of the temporal expressions found in the sentences which contain the messages and alter their referring time, as well as the normalization of those temporal expressions in relation to the publication time of the document which contains the messages. For this sub-stage we adopted a module which was developed earlier. As was mentioned earlier in this paper, the normalized temporal expressions alter the referring time of the messages, an information which we use during the extraction of the Synchronic and Diachronic Relations (SDRs).

Relations Extraction. The extraction of the SDRs is a rather straightforward process. As we can see from figure 3, once the messages have been identified and placed in the appropriate position in the grid, then in order to identify the SDRs we have simply to apply the rules for each of the relations. The results of this stage are shown in table 2.

Table 2. Evaluation for the message extraction module and the relation extraction module.

	Messages	SDRs
Precision	42.96%	30.66%
Recall	35.91%	49.12%
F-Measure	39.12%	37.76%

5 Conclusions and Future Work

In this paper we have presented a methodology which constitutes the first step towards the creation of multi-document summaries of evolving events and we have presented a concrete implementation of this proposed methodology in a case study of five topics of terroristic incidents which include hostages. The end result of the implementation is the creation of a graph whose nodes are the messages and whose vertices are the Synchronic and Diachronic Relations that connect those messages. According to [12] the architecture of a Natural Language Generation (NLG) system is composed of three main stages: (a) Document Planning, (b) Micro-Planning and (c) Surface Generation. As we have shown earlier [1, 4] we view the creation of the grid as constituting the Document Planning stage of an NLG system. We are currently working on the implementation of the other two stages (Micro Planning and Surface Generation) in order to create the final textual summaries. Of particular concern to us is the fact that in order to create the specifications of the messages and SDRs a corpus study by human beings is involved, something which adds to cost and reduces the flexibility of our approach. We are currently actively working on a methodology which will enable us to automatically create message specifications. At the core of the theory that we are working on is the fact that almost all of the observed messages are describing *actions*. Thus, each message is triggered by a set of semantically related verbs or verbalized nouns. In addition, most of the entities involved in a message (the message's arguments) are found in near proximity. Based on the above two basic remarks we are currently working on a method for automatically providing messages' specifications from raw text.

References

1. S. D. Afantenos. Some reflections on the task of content determination in the context of multi-document summarization of evolving events. In G. Angelova, K. Bontcheva, R. Mitkov, N. Nicolov, and N. Nikolov, editors, *Recent Advances in Natural Language Processing (RANLP 2007)*, pages 12–16, Borovets, Bulgaria, Sept. 2007. INCOMA.
2. S. D. Afantenos, I. Doura, E. Kapellou, and V. 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.

3. S. D. Afantenos, V. Karkaletsis, and P. Stamatopoulos. Summarizing reports on evolving events; part i: Linear evolution. In G. Angelova, K. Bontcheva, R. Mitkov, N. Nicolov, and N. Nikolov, editors, *Recent Advances in Natural Language Processing (RANLP 2005)*, pages 18–24, Borovets, Bulgaria, Sept. 2005. INCOMA.
4. S. D. Afantenos, V. Karkaletsis, P. Stamatopoulos, and C. Halatsis. Using synchronic and diachronic relations for summarizing multiple documents describing evolving events. *Journal of Intelligent Information Systems*, 2008. Accepted for Publication.
5. S. D. Afantenos, K. Liontou, M. Salapata, and V. Karkaletsis. An introduction to the summarization of evolving events: Linear and non-linear evolution. In B. Sharp, editor, *Proceedings of the 2nd International Workshop on Natural Language Understanding and Cognitive Science, NLUCS 2005*, pages 91–99, Maiami, Florida, USA, May 2005. INSTICC Press.
6. J. Allan. Introduction to topic detection and tracking. In J. Allan, editor, *Topic Detection and Tracking: Event-Based Information Organization*, chapter 1, pages 1–16. Kluwer Academic Publishers, 2002.
7. J. Allan, J. G. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic detection and tracking pilot study: Final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218, Feb. 1998.
8. D. Jones, T. Bench-Capon, and P. Visser. Methodologies for ontology development. In *Proceedings of the IT&KNOWS Conference, XV IFIP World Computer Congress*, Budapest, 1998.
9. M. F. Lopez. Overview of methodologies for building ontologies. In *Proceedings of the Workshop on Ontologies and Problem-Solving Methods: Lessons Learned and Future Trends (IJCAI99)*, Stockholm, 1999.
10. R. Papka. *On-line New Event Detection, Clustering and Tracking*. PhD thesis, Department of Computer Science, University of Massachusetts, 1999.
11. H. S. Pinto and J. P. Martins. Ontologies: How can they be built? *Knowledge and Information Systems*, 6(4):441–464, 2004.
12. E. Reiter and R. Dale. *Building Natural Language Generation Systems*. Studies in Natural Language Processing. Cambridge University Press, 2000.