Exploiting Structure for Event Discovery Using the MDI Algorithm

Martina Naughton
School of Computer Science & Informatics
University College Dublin
Ireland
martina.naughton@ucd.ie

Abstract

Effectively identifying events in unstructured text is a very difficult task. This is largely due to the fact that an individual event can be expressed by several sentences. In this paper, we investigate the use of clustering methods for the task of grouping the text spans in a news article that refer to the same event. The key idea is to cluster the sentences, using a novel distance metric that exploits regularities in the sequential structure of events within a document. When this approach is compared to a simple bag of words baseline, a statistically significant increase in performance is observed.

1 Introduction

Accurately identifying events in unstructured text is an important goal for many applications that require natural language understanding. There has been an increased focus on this problem in recent years. The Automatic Content Extraction (ACE) program[1] is dedicated to developing methods that automatically infer meaning from language data. Tasks include the detection and characterisation of Entities, Relations, and Events. Extensive research has been dedicated to entity recognition and binary relation detection with significant results (Bikel et al., 1999). However, event extraction is still considered as one of the most challenging tasks because an individual event can be expressed by several sentences (Xu et al., 2006).

In this paper, we primarily focus on techniques for identifying events within a given news article. Specifically, we describe and evaluate clustering methods for the task of grouping sentences in a news article that refer to the same event. We generate sentence clusters using three variations of the well-documented Hierarchical Agglomerative Clustering (HAC) (Manning and Schütze, 1999) as a baseline for this task. We provide convincing evidence suggesting that inherent structures exist in the manner in which events appear in documents. In Section 3.1, we present an algorithm which uses such structures during the clustering process and as a result a modest increase in accuracy is observed.

Developing methods capable of identifying all types of events from free text is challenging for several reasons. Firstly, different applications consider different types of events and with different levels of granularity. A change in state, a horse winning a race and the race meeting itself can be considered as events. Secondly, interpretation of events can be subjective. How people understand an event can depend on their knowledge and perspectives. Therefore in this current work, the type of event to extract is known in advance. As a detailed case study, we investigate event discovery using a corpus of news articles relating to the recent Iraqi War where the target event is the *"Death"* event type. Figure 1 shows a sample article depicting such events.

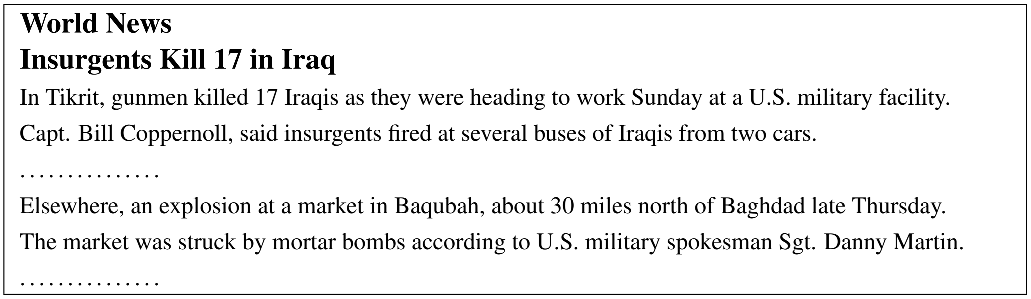
The remainder of this paper is organised as follows: We begin with a brief discussion of related work in Section 2. We describe our approach to Event Discovery in Section 3. Our techniques are experimentally evaluated in Section 4. Finally, we conclude with a discussion of experimental observations and opportunities for future work in Section 5.

2 Related Research

The aim of Event Extraction is to identify any instance of a particular class of events in a natural

[1] http://www.nist.gov/speech/tests/ace/

Proceedings of the ACL 2007 Student Research Workshop, pages 31–36,
Prague, June 2007. ©2007 Association for Computational Linguistics



World News
Insurgents Kill 17 in Iraq

In Tikrit, gunmen killed 17 Iraqis as they were heading to work Sunday at a U.S. military facility.

Capt. Bill Coppernoll, said insurgents fired at several buses of Iraqis from two cars.

. . . .

Elsewhere, an explosion at a market in Baqubah, about 30 miles north of Baghdad late Thursday.

The market was struck by mortar bombs according to U.S. military spokesman Sgt. Danny Martin.

. . . .

Figure 1: Sample news article that describes multiple events.

language text, extract the relevant arguments of the event, and represent the extracted information into a structured form (Grishman, 1997). The types of events to extract are known in advance. For example, *"Attack"* and *"Death"* are possible event types to be extracted. Previous work in this area focuses mainly on linguistic and statistical methods to extract the relevant arguments of a event type. Linguistic methods attempt to capture linguists knowledge in determining constraints for syntax, morphology and the disambiguation of both. Statistical methods generate models based in the internal structures of sentences, usually identifying dependency structures using an already annotated corpus of sentences. However, since an event can be expressed by several sentences, our approach to event extraction is as follows: First, identify all the sentences in a document that refer to the event in question. Second, extract event arguments from these sentences and finally represent the extracted information of the event in a structured form.

Particularly, in this paper we focus on clustering methods for grouping sentences in an article that discuss the same event. The task of clustering similar sentences is a problem that has been investigated particularly in the area of text summarisation. In SimFinder (Hatzivassiloglou et al., 2001), a flexible clustering tool for summarisation, the task is defined as finding text units (sentences or paragraphs) that contain information about a specific subject. However, the text features used in their similarity metric are selected using a Machine Learning model.

3 Identifying Events within Articles

We treat the task of grouping together sentences that refer to the same event(s) as a clustering problem. As a baseline, we generate sentence clusters using average-link, single-link and complete-link Hierarchical Agglomerative Clustering. HAC initially assigns each data point to a singleton cluster, and repeatedly merges clusters until a specified termination criteria is satisfied (Manning and Schütze, 1999). These methods require a similarity metric between two sentences. We use the standard cosine metric over a bag-of-words encoding of each sentence. We remove stopwords and stem each remaining term using the Porter stemming algorithm (Porter, 1997). Our algorithms begin by placing each sentence in its own cluster. At each iteration we merge the two closest clusters. A fully-automated approach must use some termination criteria to decide when to stop clustering. In experiments presented here, we adopt two manually supervised methods to set the desired number of clusters (k): "correct" k and "best" k. "Correct" sets k to be the actual number of events. This value was obtained during the annotation process (see Section 4.1). "Best" tunes k so as to maximise the quality of the resulting clusters.

3.1 Exploiting Article Structure

Our baseline ignores an important constraint on the event associated with each sentence: the position of the sentence within the document. Documents consist of sentences arranged in a linear order and nearby sentences in terms of this ordering typically refer to the same topic (Zha, 2002). Similarly we assume that adjacent sentences are more likely to refer to the same event, later sentences are likely to introduce new events, etc. In this Section, we describe an algorithm that exploits this document structure during the sentence clustering process.

The basic idea is to learn a model capable of capturing document structure, i.e. the way events are reported. Each document is treated as a sequence of labels (1 label per sentence) where each label represents the event(s) discussed in that sentence. We define four generalised event label types: N, represents a new event sentence; C, represents a continuing event sentence (i.e. it discusses the same event as the preceding sentence); B, represents a back-reference to an earlier event; X, represents a sentence that does not reference an event. This model takes the form of a Finite State Automaton (FSA) where:

- *States* correspond to event labels.
- *Transitions* correspond to adjacent sentences that mention the pair of events.

More formally, E = (S, s_0, F, L, T) is a model where S is the set of states, $s_0 \in S$ is the initial state, $F \subseteq S$ is the set of final states, L is the set of edge labels and $T \subseteq (S \times L) \times S$ is the set of transitions. We note that it is the responsibility of the learning algorithm to discover the correct number of states.

We treat the task of discovering an event model as that of learning a regular grammar from a set of positive examples. Following Golds research on learning regular languages (Gold, 1967), the problem has received significant attention. In our current experiments, we use Thollard et al's MDI algorithm (Thollard et al., 2000) for learning the automaton. MDI has been shown to be effective on a wide range of tasks, but it must be noted that any grammar inference algorithm could be substituted.

To estimate how much sequential structure exists in the sentence labels, the document collection was randomly split into training and test sets. The automaton produced by MDI was learned using the training data, and the probability that each test sequence was generated by the automaton was calculated. These probabilities were compared with those of a set of random sequences (generated to have the same distribution of length as the test data). The probabilities of event sequences from our dataset and the randomly generated sequences are shown in Figure 2. The test and random sequences are sorted by probability. The vertical axis shows the rank in each sequence and the horizontal axis shows the negative log probability of the sequence at each rank. The data suggests that the documents are indeed structured, as real document sequences tend to be much more likely under the trained FSA than randomly generated sequences.

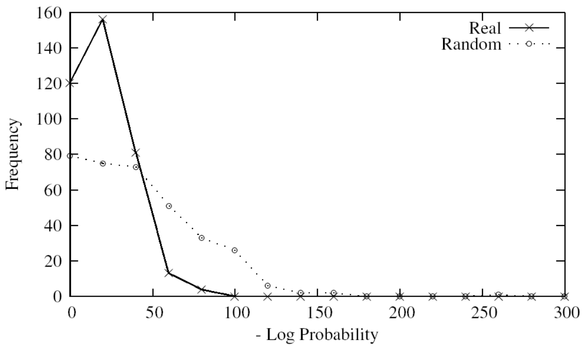


Figure 2: Distribution in the probability that actual and random event sequences are generated by the automaton produced by MDI.

We modify our baseline clustering algorithm to utilise the structural information omitted by the automaton as follows: Let $L(c_1, c_2)$ be a sequence of labels induced by merging two clusters c_1 and c_2. If $P(\mathrm{L}(c_1, c_2))$ is the probability that sequence $L(c_1, c_2)$ is accepted by the automaton, and let $\cos(c_1, c_2)$ be the cosine distance between c_1 and c_2. We can measure the similarity between c_1 and c_2 as:

$$SIM(c_1, c_2) = \cos(c_1, c_2) \times P(\mathrm{L}(c_1, c_2)) \quad (1)$$

Let r be the number of clusters remaining. Then there are $\frac{r(r-1)}{2}$ pairs of clusters. For each pair of clusters c_1,c_2 we generate the resulting sequence of labels that would result if c_1 and c_2 were merged. We then input each label sequence to our trained FSA to obtain the probability that it is generated by the automaton. At each iteration, the algorithm proceeds by merging the most similar pair according to this metric. Figure 3 illustrates this process in more detail. To terminate the clustering process, we adopt either the “correct” k or “best” k halting criteria described earlier.

4 Experiments

4.1 Experimental Setup

In our experiments, we used a corpus of news articles which is a subset of the Iraq Body Count (IBC)

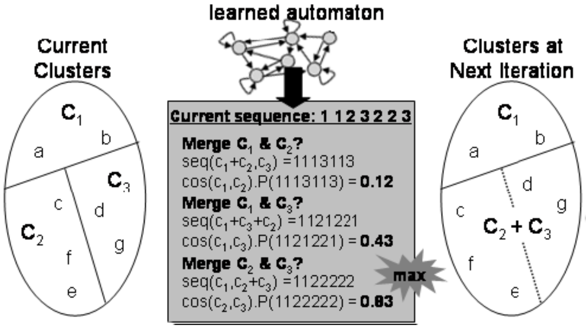


Figure 3: The sequence-based clustering process.

dataset[2]. This is an independent public database of media-reported civilian deaths in Iraq resulting directly from military attack by the U.S. forces. Casualty figures for each event reported are derived solely from a comprehensive manual survey of online media reports from various news sources. We obtained a portion of their corpus which consists of 342 new articles from 56 news sources. The articles are of varying size (average sentence length per document is 25.96). Most of the articles contain references to multiple events. The average number of events per document is 5.09. Excess HTML (image captions etc.) was removed, and sentence boundaries were identified using the Lingua::EN::Sentence perl module available from CPAN[3].

To evaluate our clustering methods, we use the definition of precision and recall proposed by (Hess and Kushmerick, 2003). We assign each pair of sentences into one of four categories: (i) clustered together (and annotated as referring to the same event); (ii) not clustered together (but annotated as referring to the same event); (iii) incorrectly clustered together; (iv) correctly not clustered together. Precision and recall are thus found to be computed as $P = \frac{a}{a+c}$ and $R = \frac{a}{a+b}$, and $F1 = \frac{2PR}{P+R}$.

The corpus was annotated by a set of ten volunteers. Within each article, events were uniquely identified by integers. These values were then mapped to one of the four label categories, namely "N", "C", "X", and "B". For instance, sentences describing previously unseen events were assigned a new integer. This value was mapped to the label category "N" signifying a new event. Similarly, sentences referring to events in a preceding sentence were assigned the same integer identifier as that assigned to the preceding sentence and mapped to the label category "C". Sentences that referenced an event mentioned earlier in the document but not in the preceding sentence were assigned the same integer identifier as that sentence but mapped to the label category "B". Furthermore, If a sentence did not refer to any event, it was assigned the label 0 and was mapped to the label category "X". Finally, each document was also annotated with the distinct number of events reported in it.

In order to approximate the level of inter-annotation agreement, two annotators were asked to annotate a disjoint set of 250 documents. Inter-rater agreements were calculated using the kappa statistic that was first proposed by (Cohen, 1960). This measure calculates and removes from the agreement rate the amount of agreement expected by chance. Therefore, the results are more informative than a simple agreement average (Cohen, 1960; Carletta, 1996). Some extensions were developed including (Cohen, 1968; Fleiss, 1971; Everitt, 1968; Barlow et al., 1991). In this paper the methodology proposed by (Fleiss, 1981) was implemented. Each sentence in the document set was rated by the two annotators and the assigned values were mapped into one of the four label categories ("N", "C", "X", and "B"). For complete instructions on how kappa was calculated, we refer the reader to (Fleiss, 1981). Using the annotated data, a kappa score of 0.67 was obtained. This indicates that the annotations are somewhat inconsistent, but nonetheless are useful for producing tentative conclusions.

To determine why the annotators were having difficulty agreeing, we calculated the kappa score for each category. For the "N", "C" and "X" categories, reasonable scores of 0.69, 0.71 and 0.72 were obtained respectively. For the "B" category a relatively poor score of 0.52 was achieved indicating that the raters found it difficult to identify sentences that referenced events mentioned earlier in the document. To illustrate the difficulty of the annotation task an example where the raters disagreed is depicted in Figure 4. The raters both agreed when assigning labels to sentence 1 and 2 but disagreed when assigning a label to Sentence 23 . In order to correctly annotate this sentence as referring to the event de-

[2] http://iraqbodycount.org/

[3] http://cpan.org/

Sentence 1: A suicide attacker set off a bomb that tore through a funeral **tent** jammed with Shiite mourners Thursday.
Rater 1: label=1. **Rater 2:** label=1
Sentence 2: The explosion, in a working class neighbourhood of **Mosul**, destroyed **the tent** killing **nearly 50 people**.
Rater 1: label=1. **Rater 2:** label=1.
....
Sentence 23: At the hospital of this **northern city**, doctor Saher Maher said that **at least 47 people** were killed.
Rater 1: label=1. **Rater 2: label=2**.

Figure 4: Sample sentences where the raters disagreed.

Algorithm	a-link	c-link	s-link
BL(correct k)	40.5 %	39.2%	39.6%
SEQ(correct k)	47.6%*	45.5%*	44.9%*
BL(best k)	52.0%	48.2%	50.9%
SEQ(best k)	61.0%*	56.9%*	58.6%*

Table 1: % F1 achieved using average-link (a-link), complete-link (c-link) and single-link (s-link) variations of the baseline and sequence-based algorithms when the correct and best k halting criteria are used. Scores marked with * are statistically significant to a confidence level of 99%.

scribe in sentence 1 and 2, the rater have to resolve that “the northern city” is referring to “Mosul” and that “nearly 50” equates to “at least 47”. These and similar ambiguities in written text make such an annotation task very difficult.

4.2 Results

We evaluated our clustering algorithms using the F1 metric. Results presented in Table 1 were obtained using 50:50 randomly selected train/test splits averaged over 5 runs. For each run, the automaton produced by MDI was generated using the training set and the clustering algorithms were evaluated using the test set. On average, the sequence-based clustering approach achieves an 8% increase in F1 when compared to the baseline. Specifically the average-link variation exhibits the highest F1 score, achieving 62% when the “best” k termination method is used.

It is important to note that the inference produced by the automaton depends on two values: the threshold α of the MDI algorithm and the amount of label sequences used for learning. The closer α is to 0, the more general the inferred automaton becomes. In an attempt to produce a more general automaton, we chose $\alpha = 0.1$. Intuitively, as more training data is used to train the automaton, more accurate inferences are expected. To confirm this we calculated the %F1 achieved by the average-link variation of the method for varying levels of training data. Overall, an improvement of approx. 5% is observed as the percentage training data used is increased from 10% to 90%.

5 Discussion

Accurately identifying events in unstructured text is a very difficult task. This is partly because the description of an individual event can spread across several sentences. In this paper, we investigated the use of clustering for the task of grouping sentences in a document that refer to the same event. However, there are limitations to this approach that need to be considered. Firstly, results presented in Section 4.2 suggest that the performance of the clusterer depends somewhat on the chosen value of k (i.e. the number of events in the document). This information is not readily available. However, preliminary analysis presented in (Naughton et al., 2006) indicate that is possible to estimate this value with reasonable accuracy. Furthermore, promising results are observed when this estimated value is used halt the clustering process. Secondly, labelled data is required to train the automation used by our novel clustering method. Evidence presented in Section 4.1 suggests that reasonable inter-annotation agreement for such an annotation task is difficult to achieve. Nevertheless, clustering allows us to take into account that the manner in which events are described is not always linear. To assess exactly how beneficial this is, we are currently treating this problem as a text segmentation task. Although this is a crude treatment of the complexity of written text, it will help us to approximate the benefit (if any) of applying clustering-based techniques to this task.

In the future, we hope to further evaluate our methods using a larger dataset containing more event types. We also hope to examine the interesting possibility that inherent structures learned from documents originating from one news source (e.g. Aljazeera) differ from structures learned using documents originating from another source (e.g. Reuters). Finally, a single sentence often contains references to multiple events. For example, consider the sentence "These two bombings have claimed the lives of 23 Iraqi soldiers". Our algorithms assume that each sentence describes just one event. Future work will focus on developing methods to automatically recognise such sentences and techniques to incorporate them into the clustering process.

Acknowledgements. This research was supported by the Irish Research Council for Science, Engineering & Technology (IRCSET) and IBM under grant RS/2004/IBM/1. The author also wishes to thank Dr. Joe Carthy and Dr. Nicholas Kushmerick for their helpful discussions.

References

W. Barlow, N. Lai, and S. Azen. 1991. A comparison of methods for calculating a stratified kappa. *Statistics in Medicine*, 10:1465–1472.

Daniel Bikel, Richard Schwartz, and Ralph Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning*, 34(1-3):211–231.

Jean Carletta. 1996. Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22:249–254.

Jacob Cohen. 1960. A coeficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Jacob Cohen. 1968. Weighted kappa: Nominal scale agreement with provision for scaled disagreement or partial credit. *Psychological Bulletin*, 70.

B.S. Everitt. 1968. Moments of the statistics kappa and the weighted kappa. *The British Journal of Mathematical and Statistical Psychology*, 21:97–103.

J.L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76.

J.L. Fleiss, 1981. *Statistical methods for rates and proportions*, pages 212–36. John Wiley & Sons.

E. Mark Gold. 1967. Grammar identification in the limit. *Information and Control*, 10(5):447–474.

Ralph Grishman. 1997. Information extraction: Techniques and challenges. In *Proceedings of the seventh International Message Understanding Conference*, pages 10–27.

Vasileios Hatzivassiloglou, Judith Klavans, Melissa Holcombe, Regina Barzilay, Min-Yen Kan, and Kathleen McKeown. 2001. SIMFINDER: A flexible clustering tool for summarisation. In *Proceedings of the NAACL Workshop on Automatic Summarisation, Association for Computational Linguistics*, pages 41–49.

Andreas Hess and Nicholas Kushmerick. 2003. Learning to attach semantic metadata to web services. In *Proceedings of the International Semantic Web Conference (ISWC 2003)*, pages 258–273. Springer.

Christopher Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press.

Martina Naughton, Nicholas Kushmerick, and Joseph Carthy. 2006. Event extraction from heterogeneous news sources. In *Proceedings of the AAAI Workshop Event Extraction and Synthesis*, pages 1–6, Boston.

Martin Porter. 1997. An algorithm for suffix stripping. *Readings in Information Retrieval*, pages 313–316.

Franck Thollard, Pierre Dupont, and Colin de la Higuera. 2000. Probabilistic DFA inference using Kullback-Leibler divergence and minimality. In *Proceedings of the 17th International Conference on Machine Learning*, pages 975–982. Morgan Kaufmann, San Francisco.

Feiyu Xu, Hans Uszkoreit, and Hong Li. 2006. Automatic event and relation detection with seeds of varying complexity. In *Proceedings of the AAAI Workshop Event Extraction and Synthesis*, pages 12–17, Boston.

Hongyuan Zha. 2002. Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in Information Retrieval*, pages 113–120, New York, NY. ACM Press.