Inf Retrieval (2010) 13:132–156
DOI 10.1007/s10791-009-9113-0

Sentence-level event classification in unstructured texts

M. Naughton · N. Stokes · J. Carthy

Received: 4 September 2008 / Accepted: 19 August 2009 / Published online: 11 September 2009

Abstract The ability to correctly classify sentences that describe events is an important task for many natural language applications such as Question Answering (QA) and Text Summarisation. In this paper, we treat event detection as a sentence level text classification problem. Overall, we compare the performance of discriminative versus generative approaches to this task: namely, a Support Vector Machine (SVM) classifier versus a Language Modeling (LM) approach. We also investigate a rule-based method that uses handcrafted lists of 'trigger' terms derived from WordNet. Two datasets are used in our experiments to test each approach on six different event types, i.e., *Die*, *Attack*, *Injure*, *Meet*, *Transport* and *Charge-Indict*. Our experimental results show that the trained SVM classifier significantly outperforms the simple rule-based system and language modeling approach on both datasets: ACE (F1 66% vs. 45% and 38%, respectively) and IBC (F1 92% vs. 88% and 74%, respectively). A detailed error analysis framework for the task is also provided which separates errors into different types: *semantic*, *inference*, *continuous* and *trigger-less*.

Keywords Information extraction · Event detection · Language modeling · Machine learning

1 Introduction

Event detection is a core Natural Language Processing (NLP) task that focuses on the automatic identification and classification of various event types in text. This task has applications in automatic Question Answering (QA), Text Summarisation and more recently in the context of Semantic Web Retrieval. For example, event recognition is a core task in QA since the majority of web user questions have been found to relate to events and

M. Naughton (✉) · N. Stokes · J. Carthy
School of Computer Science and Informatics, University College Dublin, Dublin, Ireland
e-mail: martina.naughton@ucd.ie; martina.naughton@gmail.com

N. Stokes
e-mail: nicola.stokes@ucd.ie

situations in the world (Saurí et al. 2005). For complex questions such as "How many people were killed in Baghdad in March?", QA systems often rely on event detection systems to identify all relevant event instances before formulating an answer. In addition, text summarisation research has focused on the use of phrasal concepts such as events to represent text topicality in extractive and generative summarisation tasks (Li et al. 2006; Vanderwende et al. 2004; Wu 2006; Filatova and Hatzivassiloglou 2004; Daniel et al. 2003). A common conclusion of these research efforts is that an event-based approach to summarisation improves the quality of the generated summaries. More recently, there has been a lot of interest by the Semantic Web community in automatic methods for adding rich metadata to documents such as entity and event tags. For example, Reuters, the financial news giant, recently launched an API called Open Calais[1] which is capable of adding semantic markup to unstructured HTML news documents. This API can recognise *people*, *places*, *companies*, and different *event* types. There are quite a few interesting applications of this type of semantic markup, including more effective document search and result presentation.

In this paper, we investigate the use of generative and discriminate models for identifying the sentences in a document that describe one or more instance of a specified event type. For each event type used in our experiments, the task is treated as a binary text classification problem, where each sentence is either classified as one that contains an instance of that event type or as one that does not. We view this task as a filtering step in a larger pipeline NLP architecture (e.g., a QA system), which helps speed up subsequent processing by removing irrelevant, non-event sentences. Three event detection approaches are explored in this paper. Firstly, we train a Support Vector Machine (SVM) using a variety of term, lexical and additional event based features to encode each training/test instance. Secondly, we adopt a probabilistic language modeling approach that captures how descriptions of event instances in text are likely to be generated. We estimate a series of unigram models using three well-known smoothing approaches, and examine their overall behavior on classification performance.

Event classification at a sentence level is a very challenging task. For example, if the target event is *Die*, we want our system to extract sentences such as "5 people were killed in the explosion" and "A young boy and his mother were found dead on Wednesday evening". However, that classifier must also be able to detect complex cases such as: "An ambulance rushed the soldier to hospital, but efforts to save him failed" and reject instances such as "Fragmentation mines have a killing range of 100 feet". A naïve system that selects only sentences that contain terms connected with death such as 'kill', 'die' or 'execute' may correctly detect many positive instances. Nevertheless, there are instances where this approach would fail. The aim of the third event detection system investigated in this paper is to evaluate the effectiveness of such a shallow NLP approach by developing a manual rule-based system, which finds sentences connected to a target event type using a handcrafted list of 'trigger' terms found in WordNet (Miller 1995). This system is compared with the SVM and unigram language models in order to investigate how the performance of such a manual approach compares against more sophisticated supervised techniques.

We use two datasets in our experiments. The first is the ACE 2005 Multilingual Training Corpus (Walker et al. 2006) that was annotated for 33 different event types. Within the ACE data the number of instances referring to each event type is somewhat limited. For this reason we select the six types that have the highest frequency of occurrence in the data and use these in our experiments. They include the *Die*, *Attack*, *Transport*, *Meet*, *Injure* and *Charge-Indict* types. The second corpus is a collection of articles from the

[1] http://www.opencalais.com/

Iraq Body Count (IBC) database[2] manually annotated for the *Die* event type. This dataset arose from a larger project that focuses on statistical approaches for collecting fatalities statistics from unstructured news data. We use this additional corpus to augment the data used for training a classifier on the *Die* event.

In this paper, our results show that the most effective classification approach is dependent on the target event type. For 'broad' event types[3] such as *Attack* and *Transport*, the SVM appears to be the most appropriate approach. Yet for more 'specific' types such as *Charge-Indict* and *Injure*, the trigger-based classification system produces the best overall results. Finally, our experiments also demonstrate that the type of dataset used for training significantly affects the performance of our supervised approaches on this classification task. Specifically, heterogeneous datasets with a rich vocabulary tend to be more suitable for training purposes.

We make a number of contributions in this paper. First, the applicability of Machine Learning and Language Modeling based approaches for Event Classification at a sentence-level is fully investigated. Previous research has focused on event detection at either the term/phrasal level (ACE research) or the document level (TDT research); there is no prior work that examines this problem at a sentence level, a granularity which is favored by many IR, QA and Summarisation applications. Secondly, we introduce a manual rule-based system, which consistently outperforms the language modeling approach, and performs competitively with the SVM on many of the event types. Also, a thorough error analysis methodology is described in Sect. 6, which will be useful for researchers working in this area. Finally, a novel event dataset of Iraqi war articles, where all instances of the Die event type have been manually annotated at the sentence level, is introduced.[4]

The remainder of this paper is organised as follows. We begin with a brief description of background research and related work in Sect. 2. Section 3 continues with details of the datasets used in our experiments. Section 4 describes the approaches adopted for identifying sentences that describe one or more instance of a particular event type. We experimentally evaluate and compare the performance of our event detection algorithms in Sect. 5. The common errors produced by each approach are analysed in Sect. 6. Finally, in Sect. 7, we conclude with a discussion of our experimental observations and our intentions for future work.[5]

2 Background and related work

Event detection, in the context of news stories, has been an active area of research for the best part of 10 years. Many event extraction technologies have been reported in the literature. For example, event extraction systems capable of detecting disease outbreaks (Grishman et al. 2002), conflict events (King and Lowe 2003; Atkinson et al. 2008), and natural disaster events in a multilingual setting (Atkinson et al. 2008) have been investigated. Moreover, the NIST sponsored Topic Detection and Tracking (TDT) project, which

[2] http://www.iraqbodycount.org/database/

[3] We use the term 'broad' in this context to describe events that can be described by many synonymous, near-synonymous and related 'trigger' terms. In contrast, 'specific' event types are expressed using a more limited set of vocabulary terms. These concepts are described in more detail in Sect. 5.2.2.

[4] For a copy of the IBC dataset, please contact the first author.

[5] Some initial experiments by the authors on this work are published in (Naughton et al. 2008). However, substantially more experimental results and analyses are presented in this publication.

began in 1998 investigated the development of technologies that could detect novel events in segmented or unsegmented news streams, and track the progression of these events over time (Yang et al. 1998, 1999; Allan et al. 1998). Although this project ended in 2004, event detection is still investigated by more recently established projects such as the Automatic Content Extraction (ACE) program, and in domains outside of news text such as Biomedical Text Processing (Murff et al. 2003; Hripcsak et al. 2003).

Within the ACE program, the goal of the Event Detection and Recognition (EDR) task is to identify all event instances (as well as the attributes and participants of each instance) of a pre-specified set of event types. An ACE event is defined as a specific occurrence involving zero or more ACE entities,[6] values and time expressions. Two spans of text are used to identify each event: the event *trigger* and the event *mention*. An event *trigger* or *anchor* is the word that most clearly expresses the event's occurrence. In many cases, this will be the main verb in the event mention. It can also appear as a noun "The **meeting** lasted 5 hours", or an adjective "the **dead** men...". The event *mention* is the sentence that describes the event. Even though the task of identifying event mentions is not directly evaluated in ACE, systems still need to identify them so that the various attributes and participants within the mention can be extracted. The algorithms evaluated in this paper can also be applied to the detection of event mentions that contain ACE events. Overall five sites participated in this task in 2005: University of Amsterdam, BBN Technologies, IBM, Lockheed Martin and New York University. The most similar work to that describe in this paper is detailed in (Ahn 2006), where the task is treated as a word classification problem which involves finding all the event triggers and tagging them with the relevant event label (33 event types and a 'none' event type were defined). Features used included various lexical, WordNet, dependency and related entity features. However in this work, event detection is carried out at a sentence level rather than at a term level. Therefore, no direct comparison with previously published ACE results is possible.

Much research regarding Event Detection in unstructured texts came about as a result of the TDT initiative. For instance, the aim of the First Story Detection (FSD) or New Event Detection (NED) task (as it is also known) was to flag documents that discuss breaking news stories as they arrive on a news stream. Dragon Systems adopted a LM approach to this task (Allan et al. 1998; Yamron et al. 2002), building discriminator topic models from the collection and representing documents using unigram term frequencies. They then employed a single-pass clustering algorithm to identify documents that describe new events (i.e., all seed documents that form new clusters). The overall goal of the TDT Event Tracking task was to track the development of specific events over time. A number of information retrieval and machine learning techniques have been investigated for this task, including k-Nearest Neighbour (kNN) classification, Decision Tree induction and a variety of LM approaches (Allan et al. 1998; Yang et al. 1998, 1999, 2000; Walls et al. 1999; Schultz and Liberman 1999; Schwartz et al. 1997). However, these TDT tasks were somewhat restrictive in the sense that detection is carried out at a document level.

Our work differs from previous TDT and ACE research since this event detection task is performed at a sentence level where the amount of data available for building discriminating event models is far more limited. Although very little research has focused on event detection at a sentence level some work has been carried out in similar text classification problems at this level of granularity. For instance, the vast majority of email classification systems (such as spam detection (Sahami et al. 1998; Segal et al. 2004) and automatic

[6] An ACE Entity is an entity identified using guidelines outlined by EDR task, see http://projects.ldc.upenn.edu/ace/annotation for more details.

foldering (Segal and Kephart 2000; Aery and Chakravarthy 2004; Dredze et al. 2006) systems) have employed text classification techniques such as naïve bayes, rule learners, and SVMs. Also, summarisation systems often rely on methods of extracting useful sentences to include in an end summary. (Allan et al. 2001) treated this very task as a sentence-level text classification problem. Specifically, they adopted Language Modeling techniques to find the sentences in a document that are both novel and relevant to a given topic being summarised. A year after its publication, this seminal research went on to motivate the introduction of the Novelty Detection Track at TREC.[7] Due to the success of this approach, coupled with the fact that language modeling approaches have been applied to many other classification tasks (Lewis 1992; Larkey and Croft 1996; Sahami 1996; Sahami et al. 1998; McCallum and Nigam 1998), we investigate the applicability of similar language modeling based techniques for the event classification task described in this paper.

3 Corpora

The ACE 2005 Multilingual Corpus was annotated for *entities*, *relations* and *events*. It consists of articles originating from six difference genres including Newswire (20%), Broadcast News (20%), Broadcast Conversation (15%), Weblogs (15%), Usenet Newsgroups (15%) and Conversational Telephone Speech (15%). We evaluate our methods on the following event types which have a high number of instances in the collection: *Die*, *Attack*, *Transport*, *Meet*, *Injure* and *Charge-Indict*.

The data we use from the IBC database consists of Newswire articles gathered from 77 different news sources. To obtain a gold standard set of annotations for articles in the IBC corpus, we asked ten volunteers to mark up all *Die* event instances. To maintain consistency across both datasets, events in the IBC corpus were identified in a manner that conformed to the ACE annotation guidelines. In order to approximate the level of inter-annotation agreement achieved for the IBC corpus, 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). Using the annotated data, a kappa score of 0.67 was obtained, indicating that while the task is difficult for humans the data is still useful for our training and test purposes. Discrepancies were adjudicated and resolved by an independent volunteer. Statistics describing both datasets are listed in Table 1.

When the IBC and ACE datasets are compared, we find that there are properties that differ between them. For instance, the IBC corpus consists only of newswire articles whereas the ACE data is made up from different genres of documents such as weblogs, news articles and Usenet newsgroups. As a result, the vocabulary used to describe the *Die* event for example in the ACE data is more diverse as its event instances occur across more topics (e.g., wars, natural disasters, traffic accidents, murders, terrorist attacks etc.). In contrast, the IBC data only contains *Die* events that occur in the context of the recent Iraqi war (see Fig. 1 for specific examples). This greater diversity in vocabulary within topics is also clear when we compare the term statistics of the ACE and IBC datasets. For example, although the average number of total terms per document is larger in the IBC corpus (585.92) than the ACE data (445.55), the average number of unique terms per document is

[7] The TREC novelty track ran from 2002 to 2004. The aim of the task was to highlight sentences containing relevant and new information in a short, topical document stream. This is analogous to highlighting key parts of a document for another person to read, and this kind of output can be useful as input to a summarization system.

Table 1 Statistics describing both datasets where **Trans**. and **Charge** refer to the *Transport* and *Charge-Indict* event types, respectively

	ACE						IBC
	Die	Injure	Attack	Meet	Trans.	Charge	Die
No. of sentences	4496	1487	6962	2639	5934	637	8628
No. of documents	154	50	235	84	181	43	332
No. Ev. Sent.	392	87	984	160	472	85	2262
Avg. Doc. length	29.2	29.7	29.6	31.4	32.8	14.8	25.9
Avg. Ev. Sent./Doc.	2.54	1.74	4.18	1.92	2.55	2.60	6.78

IBC *Die* instances
Seven US soldiers were **killed** when their vehicle hit an explosive device in Baghdad.
Four Iraqi soldiers and three civilians were also **killed** in separate attacks in the northern Iraqi city of Arbil.
In the northern city of Mosul, bodies of 18 Iraqis were found **dead** on Tuesday.

ACE *Die* instances
As you know about a month ago Peterson's body **washed up** right here.
Of all those who have been **killed**, at least 19 of them were said to be victims of friendly or accidental fire.
Who are you to tell him that his suffering is only worth $250 000 for the **loss of a child**?

Fig. 1 Sample *Die* event instances taken from articles in the ACE 2005 and IBC datasets

much higher in ACE (49.8) than IBC (32.4). Another reason for this diversity, is that ACE articles contain a combination of informal (e.g., from weblogs and Usenet newsgroups texts) and formal reporting vocabulary (e.g., newswire). Based on this analysis we conclude that the ACE dataset is a heterogeneous event corpus, while the IBC dataset is a homogeneous one.

4 Event detection as classification

In this paper, we treat event detection as a series of binary classification tasks, one for each event type used in our experiments. It is important to emphasise that event detection is not treated as a multi-class classification problem in this paper. Instead, a binary classifier is built for each event type such that a sentence belonging to the data collection of that type is assigned to one of the following classes:

- An **On-Event Sentence** is a sentence that contains one or more instance of the target event type.
- An **Off-Event Sentence** is a sentence that does not contain any instance of the target event type.

4.1 A machine learning approach

In an attempt to develop an appropriate classification approach for this task we use an SVM to automatically classify each instance as either an *on-event* or *off-event* sentence. SVMs have been shown to be more robust in classification tasks involving text where the dimensionality is high (Joachims 1998). In our experiments we used a relatively efficient implementation of an SVM called the Sequential Minimal Optimisation (SMO) algorithm (Platt 1999) that is provided through the Weka framework (Witten and Frank 2000). One advantage of using this implementation is that the amount of memory required by SMO is linear to the size of the data. When an SVM is used for classification, it is important that an appropriate kernel function is chosen. For classification tasks such as this, where the number of features is large, it has been reported (Hsu et al. 2000) that a linear kernel is typically the most suitable. To confirm this, we experiment with three kernel types, namely polynomial, RBF and linear kernels. Also, parameter optimisation for the *C* penalty parameter was performed across all kernels during the cross validation process. The effects of this optimisation process together with the effects of altering the SVM's kernel type are reported in Sect. 5.2.

For difficult NLP tasks such as QA, more complex feature representations (beyond the standard bag of words) have been proposed (Moschitti et al. 2007; Surdeanu et al. 2008) recently. However, such representations, although suited to the QA task require expensive pre-processing and have never been shown to yield significant improvements in performance for classification tasks such as the one investigated in this paper. Instead each sentence forms a training/test instance for our classifier and is encoded using the following set of features:

– **Terms:** Stemmed terms (using Porter's stemming algorithm (Porter 1997)), with a frequency in the training data greater than two, were used as term features. All stopwords were removed from this feature set.
– **Lexical Information:** The presence or absence of each part of speech (POS) tag and chunk tag was used as a feature. We used the Maximum Entropy POS tagger and chunker, provided with the C&C Toolkit (Curran et al. 2007). The POS tagger uses the standard set of grammatical categories from the Penn Treebank and the chunker recognises the standard set of grammatical chunk tags: *NP* (Noun Phrase), *VP* (Verb Phrase), *PP* (Prepositional Phrase), *ADJP* (Adjective Phrase), *ADVP* (Adverb Phrase) and so on. Chunk tags are used widely within the Computational Linguistics community to represent phrasal-level clauses in a span of text. For example, if a sentence contains any noun phrase, its corresponding *NP* chunk feature would be assigned the value '1'. Otherwise, if no noun phrase were present, the value assigned to this feature would be '0'.
– **Noun Chunks:** Noun chunks with a frequency greater than two were also used as a feature. Examples include 'American soldier' and 'suicide bomb'.
– **Additional:** We added the following additional features to the feature vector: sentence length, sentence position, presence/absence of negative terms (e.g., no, not, didn't, don't, isn't, hasn't), presence/absence of a modal terms (e.g., may, might, shall, should, must, will) and the presence/absence of a location, person, organisation and a time-stamp. Named Entities are recognised using the named entity identifier available in the C&C Toolkit. Time-stamps were identified using in-house software developed by members of the Language Technology research group at the University Melbourne.[8] Our belief is that these additional features will aid the learner to correctly identify *on-event* sentences of the target event. For example, intuitively sentences at the beginning of a document are more likely to be *on-event* sentences since the lead sentences of a

[8] http://www.cs.mu.oz.au/research/lt/

document are often used to describe the major events discussed in the article. Therefore, we expect that the ‘sentence position’ feature will prove useful for this task. We evaluate the overall effectiveness of these additional features in Sect. 5.2

In the past, feature selection methods have been found to have a positive effect on classification accuracy of text classification tasks. To examine the effects of such techniques on this particular task, we use Information Gain (IG) to reduce the number of features used by the classifier by 50%. For example, if 400 hundred features were used to encode each instance originally, IG would select the top 200 most discriminative features and utilize this latter set to represent each train/test instance. While there are several motivating reasons for using IG for feature selection, there are also many limitations associated with it. One problem is that it tends to show unfair favouritism toward attributes with a large number of possible values (Tang and Liu 2005). That is, it is likely to prefer to split on an attribute with thirty possible values than one with only two possible values. In light of this drawback, the Gain Ratio metric introduced by (Quinlan 1993) is a preferred feature selection metric, which is designed to address such biases toward high-valued attributes. Although the majority of the features used for this task have only 2–3 possible values, we experiment with the IG and Gain Ratio metrics and report their effects on the SVMs overall performance in Sect. 5.2. The use of wrapper based techniques (such as Forward Selection or Backward Elimination), or alternative techniques such as random forests could be used instead of IG or Gain Ratio, and might produce some performance improvement. However, the improvement would not be expected to be very significant. For a full review of possible feature selection techniques that could be used, we refer the reader to (Cunningham 2008).

4.2 Language modeling

The language modeling approach presented in this section is based on Bayesian decision theory. Consider the situation where we wish to classify a sentence s_k into a category $c \in C = \{C_1 \ldots C_{|C|}\}$. One way to do this is to choose the category that has the largest posterior probability given the training text:

$$c^* = \arg\max_{c \in C}\{Pr(c|s_k)\} \tag{1}$$

Using Bayes rule, this can be re-written as:

$$\begin{aligned} c^* &= \arg\max_{c \in C}\{Pr(s_k|c)\,Pr(c)\} && (2)\\ &= \arg\max_{c \in C}\{Pr(s_k|c)\} && (3)\\ &= \arg\max_{c \in C}\left\{\prod_{i=1}^{N} Pr_c(w_i)\right\} && (4)\end{aligned}$$

where deducing Eq. 3 from Eq. 2 assumes uniform weighted categories. Here, $Pr(Pr(s_k|c))$ is the likelihood of s_k under category c, which can be computed using Eq. 4. Therefore, for each event type, we construct a unigram language model $LM(c_i)$, for each possible class c_i (i.e., the *on-event category* and the *off-event category*) using sentences from the training data belonging to that category. Then to classify a new sentence s_k, we supply s_k to each model, and assign it to the winning category according to Eq. 3.

One drawback of these models is that they generally under-estimate the probability of any previously unseen word in the sentence. To combat this problem smoothing techniques are used to assign a non-zero probability to the unseen words and as a result improve the accuracy of overall term probability estimation. Many smoothing methods have been proposed over the years and in general they all try to discount the probabilities of seen terms and assign the extra probability mass to the unseen words. In IR, it has been found that the choice of smoothing method affects retrieval performance (Zhai and Lafferty 2001; Kraaij and Spitters 2003). For this reason, we experiment with various smoothing techniques and compare their effects on classification performance in Sect. 5.

One of the simplest proposed solutions to this problem is the *Laplace* smoothing method (Manning and Schütze 1999). Similar to (Allan et al. 2001) we use a variant of this technique by adding 0.01 to the numerator and multiply the denominator by 1.01 as follows:

$$P_{lp}(w|LM(c_i)) = \frac{tf(w, c_i) + 0.01}{|c_i| \cdot 1.01} \tag{5}$$

By modifying the frequency of terms in this way, all unseen words either meaningful or not, will be assigned the same probability, which is not ideal. For example, consider the event type *Attack*, and two test sentences both containing terms that were not seen in the training data. Given that one of these terms is 'sand' and the other is 'knife', using the *Laplace* smoothing approach both of these unseen terms will be assigned the same likelihood of occurrence, which is counterintuitive given that the target event type is *Attack*. To overcome this problem, alternative smoothing methods exist which try to estimate the probability of unseen terms with respect to some background model. In IR applications this is usually built from the entire corpus. The idea is to attribute different probabilities to unseen words according to their global distribution in the collection. *Jelinek-Mercer* smoothing is a linear interpolation smoothing approach that does exactly this. It combines the maximum likelihood estimation (MLE) of $P(w|LM(c_i))$ from the class model with MLE of $P(w|C)$ from the collection model where C is the entire collection of documents. It uses a coefficient λ to control the influence of each model as follows:

$$P_{jm}(w|LM(c_i)) = (1 - \lambda)P(w|LM(c_i)) + \lambda P(w|C) \tag{6}$$

High values of λ lead to more smoothing. This means that the background probabilities of unseen terms have a greater influence on final term probabilities. For smaller classes this is a desirable property due to the limited number of seen terms in the training data. We experiment with varying values, where $\lambda \in [0, 1]$, and found 0.5 to be the optimal value. *Absolute Discounting* is a similar approach to smoothing where the count of each seen term is reduced by a constant δ and the discounted probability mass is redistributed amongst the unseen words in a manner which is proportional to their probability in the collection model as follows:

$$P_{ad}(w|LM(c_i)) = \frac{max(tf(w, c_i) - \delta, 0) + \delta|c_i|_u \cdot P(w|C)}{|c_i|} \tag{7}$$

where $|c_i|_u$ is the number of distinct terms in class c_i and $\delta \in [0, 1]$. Again, higher values of δ result in higher levels of smoothing. We experiment with varying values where $\delta \in [0, 1]$ and found that 0.5 proved to be the most effective value for δ, and as a result used this value in the experiments presented in Sect. 5.2.

For this task, we normalise all numeric references, locations, person names and organisations to *DIGIT*, *LOC*, *PER*, and *ORG*, respectively. This will help to reduce the dimensionality of our models, and hopefully improve their classification accuracy, particular in cases where unseen instances of these entities occur in the test data. The effect of this normalisation process is examined in Sect. 5.2.

4.3 Trigger-based event classification

According to the ACE annotation guidelines[9] event instances are identified in the text by finding event triggers that explicitly mark the occurrence of each instance. As a result, each event instance tagged in our datasets has a corresponding trigger that the annotators used to identify it. For example, terms such as ‘killing’, ‘death’ and ‘murder’ are common triggers used to identify instances of the *Die* event type. Therefore, the trigger-based event classification system selects sentences containing one or more candidate trigger terms as positive *on-event* instances. With the aid of a volunteer, we used WordNet to manually create a list of terms for the system that are synonyms, near-synonyms and related terms of the event type in question. For example, in the case of the *Meet* and *Die* events, common trigger terms include {‘encounter’, ‘visit’, ‘reunite’} and {‘die’, ‘suicide’, ‘assassination’}, respectively. We classify each sentence for a given event type as follows: if a sentence contains one or more terms in the trigger list for that event type then it is assigned to the *on-event* class for that type. Otherwise, it is assigned to the *off-event* class. Table 2 contains the number of trigger terms used for each event type.[10]

4.4 Baseline systems

We compare the performance of the trained SVM, unigram language models and the trigger-based classification system against the following baseline systems in order to assess the overall difficulty of the task:

- **Random** assigns each sentence randomly to one of the possible classes, i.e., on-event or off-event.
- **Minority Class Baseline** assigns each sentence to the class that is least frequent in the training data. In our case, this is the *on-event* class.

The next section reports the results of our experiments, the aims of which are to determine the performance of these event classification systems on some unstructured news data.

5 Evaluation methodology and results

5.1 Evaluation methodology

A standard measure for classification performance is classification accuracy. However, for corpora where the class distribution is skewed (as is the case in our datasets where approximately 90% of the instances belong to the *off-event* class) this measure can be misleading. In this section, we report *on-event* evaluation scores for each event as defined by the following metrics: Precision, Recall and F1.

Let a be the number of sentences *correctly* classified by system s as an *on-event*, b is the total number of sentences classified by s as an *on-event*, and c is the total number of human-annotated sentences in the *on-event* class. Then, the Precision, Recall and F1 score for the *on-event* class for system s can be defined as:

[9] Available at http://projects.ldc.upenn.edu/ace/annotation/

[10] Trigger term lists are available at: http://csserver.ucd.ie/~martina/triggerLists.html

Table 2 Trigger term list sizes for the six event types used in the experiments

Event type	# Triggers terms	Event type	# Triggers terms
Die	29	Transport	14
Meet	12	Injure	10
Charge-Indict	8	Attack	8

$$Precision_s = \frac{a}{b}, Recall_s = \frac{a}{c} \quad \text{and} \quad F1_s = \frac{2 \times Precision_s \times Recall_s}{Precision_s + Recall_s} \tag{8}$$

To obtain an accurate indication of each classifier's overall performance, average F1 scores across all events are often computed. These averages can be computed in two ways to reflect the importance of the smaller events. The first method, called *macro averaging*, gives an equal weight to each event and is obtained by computing an average of the F1 scores achieved. The second, called *micro averaging*, assigns weights to each event in a way that is proportional to the frequency of that event in the collection. For classes with a small amount of positive training instances, it is typically more difficult to achieve good classification, and their poor performance will have a larger effect on the overall performance when the macro average is used. Since the events in our corpora have unbalanced distributions we report both the micro and macro F1 scores.

5.2 Results

In this section, we present the results obtained for our different event classification approaches. More specifically, the purpose of the experiments reported in this section is to answer the following questions:

1. What are the effects of varying the kernel function used on the SVM's classification performance?
2. How does the performance of each approach differ across the six event types explored in this paper?
3. Do additional linguistic features improve the classification performance of the SVM?
4. How effective are the unigram language models as event classifiers?
5. What is the effect of varying the size of classifier training data?
6. What is the effect of homogeneous versus heterogeneous training data on classification performance?

5.2.1 Comparing the performance of the SVM using different kernel functions

To determine which kernel function is more suitable for this classification task, we built three versions of the SVM. The first was built using a linear kernel, the second with a polynomial kernel, and the third using an RBF kernel function. Each variation was evaluated using the IBC data where the target event type is *Die*. The resulting Precision, Recall and F1 scores achieved for the *on-event* class are contained in Table 3. To produce these results, optimisation of the C penalty parameter and gamma (RBF kernel only) was carried out. As shown, the RBF and linear kernel functions tend to outperform the polynomial kernel function. Moreover, the linear and RBF kernels tend to produce very similar results for this task. This is possibly because the linear kernel is a special case of RBF, as (Keerthi

Table 3 % Precision, Recall and F1 for the *on-event* class achieved by the SVM (with parameter optimisation), using a polynomial (SVM_{poly}), linear (SVM_{linear}) and RBF (SVM_{RBF}) kernel function

Algorithm	On-Event class		
	Precision	Recall	F1
SVM_{linear} (All Feats. IG)	91.83	92.94	92.38
SVM_{RBF} (All Feats. IG)	91.42	93.18	92.28
SVM_{poly} (All Feats. IG)	85.60	86.48	85.98

These scores are generated from the IBC dataset using 10-fold cross validation where the target event type is *Die*

and Lin 2003) show that the linear kernel with a penalty parameter C has the same performance as the RBF kernel with certain values of C. In the remainder of the experiments presented in the sections to follow, a linear kernel, with parameter optimisation is used since our results suggest that it marginally outperforms the other kernels in this classification task.

5.2.2 *Comparing event classification performance across events*

We begin this discussion by focusing on the *Die* event type since we have additional training and test data for this event from the IBC data collection. Results for the other five event types annotated in the ACE data are discussed later in this section. Table 4 shows the Precision, Recall and F1 scores achieved for the *on-event* class obtained by each approach. Two variations of the SVM using a linear kernel function were built. The first version (denoted in Table 4 by SVM_{linear} (*AllFeats. IG*)) was trained using all the features (approximately 5000) to encode each training/test instance where the features were reduced using Information Gain (IG). In the second version, the same set

Table 4 % Precision, Recall and F1 for the *on-event* class achieved by all algorithms where the target event type is *Die*

Algorithm	*On-Event* class		
	Precision	Recall	F1
SVM_{linear} (All Feats. IG)	91.83	92.94	92.38*
SVM_{linear} (All Feats.)	92.09	92.40	92.23*
Trigger-based	83.34	92.51	87.66*
LM(DS) Feature Norm	64.93	84.77	73.47
LM(DS) No Feature Norm	63.43	85.38	72.71
LM(LP) Feature Norm	61.62	84.09	71.04
LM(LP) No Feature Norm	61.28	83.78	70.72
LM(JM) Feature Norm	54.49	93.81	68.89
LM(JM) No Feature Norm	54.50	93.31	68.75
Minority Class	28.26	100.0	43.16
Random	25.48	47.88	33.15

These scores are generated from the IBC dataset using 10-fold cross validation

* indicates that the result isstatistically significantly (95% confidence level) better than the results of the systems listed in the bottom half of the table

of features was used, but no feature reduction was carried out (denoted in the table by SVM_{linear} (*AllFeats*)). *LangModel(JM)*, *LangModel(DS)* and *LangModel(LP)* represent unigram language models smoothed using *Jelinek-Mercer*, *Discount Smoothing* and *Laplace* techniques, respectively. Two variations of each language model were also built. The first variation of each model was built using feature normalisation as discussed in Sect. 4.2 (i.e., replacing all locations, organisations with common keywords such as *LOC* and *ORG*). Such models are denoted in the text as *Feature Norm*. When building the second variation of each model, no feature normalisation was used. These models are denoted in the text as *No Feature Norm*.

Overall, results in this table suggests that the SVM (built using a linear kernel function) using IG is the most effective method for correctly classifying *on-event* sentences using IBC data for the *Die* event type. Similar results were achieved when the Gain Ratio technique was used instead of IG for feature selection purposes. Since the Wilcoxon signed-rank test showed that there was no statistically significant difference between these results, we omitted them from Table 4. When feature selection techniques are not used, a marginal decrease in performance is observed. The fact that each version of the SVM obtains an *on-event* F1 score of about 90% is extremely encouraging when one considers the large skew in class distribution that is present here (i.e., the majority of training instances belong to the *off-event* class). Moreover, the Wilcoxon signed-rank test also revealed that the scores produced by both versions of the SVM are statistically significantly better (indicated by a * in Table 4) than those produced by the trigger-based system and the remaining systems listed in the bottom half of Table 4.

The results in Table 4 also indicate that the trigger-based system performs very well, achieving similar scores to the SVM. This outcome suggests that selecting sentences containing terms strongly associated with the target event is an effective way of solving this task. Nevertheless, its precision for the *on-event* class is about 8% lower than that of the SVM variations, suggesting that this approach tends to make more false alarm type mistakes, where negative instances are classified as being positive *on-events*. More specifically, many sentences that contain terms like ‘suicide’ as part of a noun phase (e.g., ‘suicide bomb’ or ‘suicide car driver’) do not report a death. The trigger-based system will place these in the *on-event* class whereas the SVM correctly places them in the *off-event* category. Finally, the Wilcoxon signed-rank test revealed that the results produced by the trigger-based system are statistically significantly better (indicated by a * in Table 4) than those produced by the systems listed in the bottom half of Table 4.

In general, the language modeling based techniques are not as effective as the SVM or trigger-based systems for the *Die* event type. Nevertheless, Table 4 shows that each language model achieves F1 scores of approximately 70% for the *on-event* class. It is also evident from this table that using the feature normalisation process discussed in Sect. 4.2 to reduce the term features used to build the language models has only marginal effects on the % *on-event* F1 score.

So far we have looked at system performance for the *Die* event on the IBC data collection. Figure 2 shows the % F1 of the *on-event* class achieved using 10-fold cross validation by all approaches on the six selected event types defined in the ACE data. To produce these results, a binary classifier (with the ability to discriminate between *on-event* and *off-event* sentences) was built for each event type.

Overall, these results indicate that the performance of each approach for the *on-event* class varies considerably across the event types. They also show that the majority of approaches perform marginally better on event types such as *Charge-Indict* and *Die*. One plausible reason for this is that these events are very ‘specific’ where only a few terms

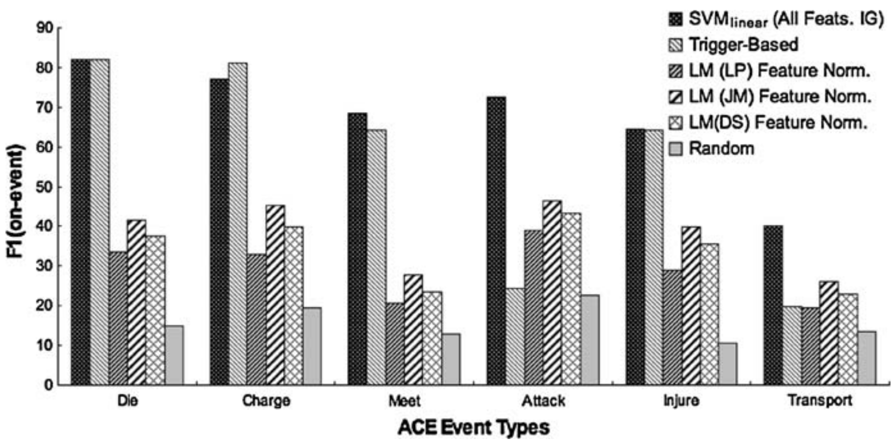


Fig. 2 % F1 of the *on-event* class achieved by each method for all six event types. These scores are generated from the ACE dataset using 10-fold cross validation

A rocket holding the first of two Mars rovers blasted off Tuesday on a seven-month **voyage** to the red planet.

I am tired, hungry, and I **leave** for work and get home from work in the dark.

I'm **taking** cars to the shop for things that are not my fault.

Wanita Renea Young, 49, filed a lawsuit complaining that the unsolicited cookies, **left** at her house after the girls knocked on her door, had triggered an anxiety attack that **sent** her to the hospital.

Fig. 3 Sample *Transport* event instances taken from articles (of varying types, i.e., weblogs, newswire etc.) in the ACE 2005 corpus

(e.g., 'charge', 'accuse', 'die', 'killed') are typically associated with their occurrences. Therefore, the SVM is able to learn these term features during training, and the trigger-based system has sufficient coverage of these terms in its trigger list. Consequently, both systems achieve their highest *on-event* F1 scores when the target event is *Charge-Indict* or *Die*. However, for broader types such as *Attack* and *Transport*, the trigger-based system performs poorly. This is probably because instances of these types are discussed in many different contexts and circumstances, and thus require a larger vocabulary to describe them. To illustrate this point further, sample instances of the *Transport* event type are shown in Fig. 3. We see from Fig. 2 that the SVM performs well on such event types, outperforming the trigger-based system on the *Attack* and *Transport* events by a percentage difference of approximately 50% and 20%, respectively.

To gain an accurate indication of each system's overall performance, the % macro and micro average F1 was calculated across the six ACE event types for all systems. These results are shown in Table 5. Overall, system rankings across the micro and macro scores remain consistent, as do corresponding scores for each system. However, this is not true in the case of the trigger-based approach; its micro average F1 score is approximately 10% lower than its macro score. The reason for this is that the trigger-based system performs very well on small events such as *Charge-Indict*, *Die* and *Meet*, but performs relatively poorer on the larger event types (*Attack* and *Transport*).

Table 5 % Macro and micro F1 across all six ACE events (i.e., *Die*, *Charge-Indict*, *Meet*, *Attack*, *Injure* and *Transport*)

Algorithm	% Micro F1	% Macro F1
SVM_{linear} (All Feats. IG)	66.37*	67.46
Trigger-based	45.32	55.93
LM(JM) Feature Norm	37.84	37.80
LM(DS) Feature Norm	34.26	33.78
LM(LP) Feature Norm	30.74	29.08
Random	16.86	15.67

* indicates that the result is statistically significantly (95% confidence level) better than the results of the systems listed in the bottom half of the table

Finally, we used Wilcoxon's signed-rank test to determine the best performing classifier based on both the macro-level and micro-level evaluations shown in Table 5. This test revealed that there was no statistically significant difference between the macro F1 scores produced by the SVM_{linear} (*AllFeats. IG*) and trigger-based systems. On the other hand, it did show that the micro F1 scores produced by the SVM_{linear} (*AllFeats. IG*) system are statistically significantly better (95% confidence level) than those generated by the trigger-based system. Finally, across both forms of evaluation we found that the results produced by the SVM_{linear} (*AllFeats. IG*) are statistically significantly better (95% confidence level) than those produced by all variations of the language models. However, Wilcoxon's signed-rank test also revealed that there is no statistically significant difference between the results produced by the trigger-based system and those produced by the language models.

5.2.3 Effectiveness of linguistic features

To access the effectiveness of each feature set on overall classification performance, we evaluated the SVM_{linear} (*AllFeats. IG*) system using different feature set combinations for all event types across both datasets using 10-fold cross validation to generate the results. These scores are shown in Fig. 4 where *terms*, *nc*, *lex*, *additional* denote the terms, noun

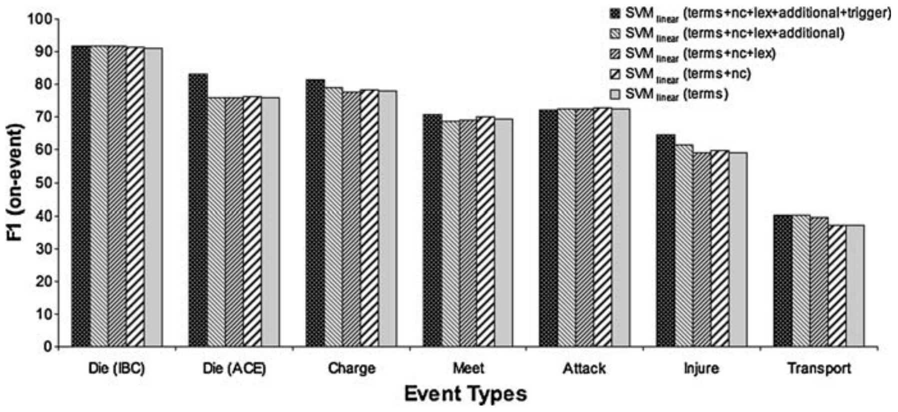


Fig. 4 % F1 for the *on-event* class achieved by the SVM using different combinations of features for the six event types using the IBC and ACE datasets, respectively. These scores are generated using 10-fold cross validation

chunks, lexical, and 'additional' features, respectively. See Sect. 4.1 for more details on how these features were constructed. Due to the success of the trigger-based system reported in the previous subsection, it also makes sense to investigate the effectiveness of the trigger term lists as a potential SVM feature. The *trigger* feature is a binary feature which is encoded as follows: if the candidate sentence contains a trigger term for the given event then a value of 1 is assigned to this feature, otherwise 0 is assigned.

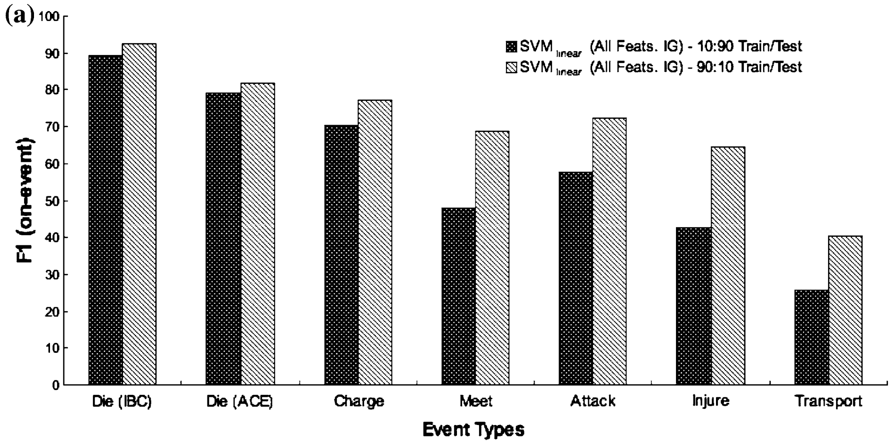
Looking at the results of these experiments in Fig. 4, we see that *term* features are the most valuable feature type for this task, since the addition of the other feature sets provides no significant increase (or decrease) in F1 score. For some event types such as *Meet*, a little improvement is obtained from adding the additional *trigger* feature, but it seems that most of the trigger terms are successfully learned by the SVM. When we examined the top 100 most discriminative individual features ranked using information gain, 60% of those included features from the *Noun Chunk*, *Lexical* and *Additional* feature sets. Specifically, the presence/absence of a numerical token, the presence/absence of the *VBD* (past tense verb) part of speech tag, sentence length and sentence position are ranked as the 2nd 4th 6th and 14th most discriminative features, respectively when the target event is *Die*. Numerical terms for example, tend to occur frequently in *Die on-event* sentences because such sentences typically include a fatality count. However, it turns out that this feature is not found to be as useful for other event types. Also, the *VBD* (part of speech tag indicating a past tense verb) part-of-speech feature is also found to be discriminative for the *Die* type, since sentences discussing an event that are written in the past tense typically signifies that the event has occurred, whereas sentences discussing an event written in the conditional may not. Unlike the numerical feature, this feature is also found to be useful across all event types. This suggests that some of these additional features are more discriminative for some event types and less discriminative for others. We feel this is an interesting result and hope to investigate this further as part of future work.

5.2.4 *Effects of varying training data size*

The graphs in Fig. 5 depict the F1 scores of the *on-event* class achieved by SVM_{linear} (*AllFeats. IG*) (Fig. 5a) and LangModel(DS) (Fig. 5b) systems for all event types using two levels of training data during the cross validation process. The *10:90 Train/Test* split was generated by using one partition of the data to train the algorithms, and the remaining nine for testing purposes. In contrast, the *90:10 Train/Test* split was generated using nine partitions of the data for training and the remaining one for testing purposes. Baring this in mind, Fig. 5a suggests that the *on-event* F1 score of all types improves by increasing the amount of data used for training. More specifically, we observe that this increase is generally greater for the ACE types. This is probably because the data for each event is quiet limited to begin with. Similarly, Fig. 5b shows analogous results when the *LangModel(DS) Feature Norm.* language model variation is used. We include these result to illustrate how the language models overall are affected by an increase in the amount of data used to build the models.

5.2.5 *Effectiveness of the language modeling approaches*

The results presented so far show that the language modeling based techniques are not as effective as the SVM approach or trigger-based system for this classification task on all event types and both datasets. Overall, models smoothed with the *Laplace* method tend to have the least impact out of the three smoothing techniques investigated. This is due to the



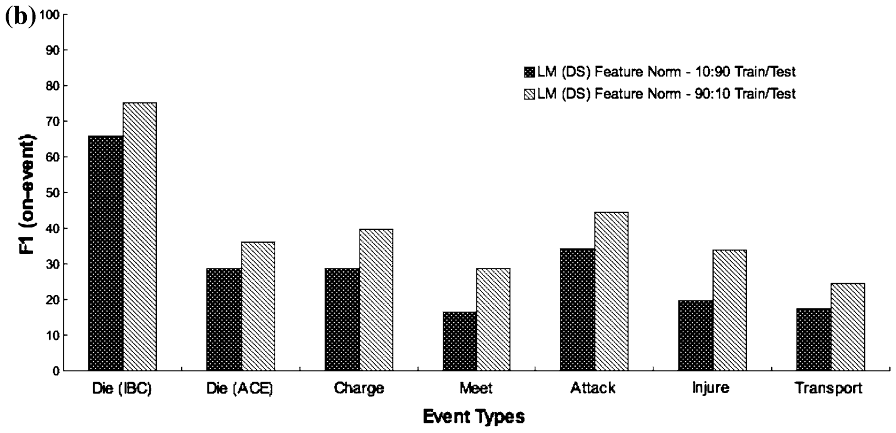


Fig. 5 % F1 for the *on-event* class depicting the effects of using varying levels of training data across all event types generated using 10-fold cross validation. **a** % F1 for the on-event class for the SVM (All Feats. IG) system using varying levels oftraining data across all event types; **b** % F1 for the on-event class for the LangModel(DS) system using varying levels of trainingdata across all event types

fact that this method assigns the same probability to all unseen terms. Thus, terms that appear frequently in the overall collection have the same likelihood of occurring in an *on-event* sentence as terms that rarely occur in the overall collection. In contrast, in the case of the *Jelinek-Mercer* and *Absolute Discounting* smoothing methods, term probabilities of unseen terms are estimated in a manner that is proportional to their global distribution in the entire corpus. Consequently, the probabilities assigned to unseen terms tend to be more reliable approximations of true term probabilities.

5.2.6 Homogeneous versus heterogeneous training data

When we compared the IBC and ACE datasets (for the *Die* event) in Sect. 3, we found that there are some properties that differ between them (e.g., the number of topics covered and the size of there respective vocabularies). We hypothesise that these differences will have a direct effect on each system's performance, particularly that of the language models since

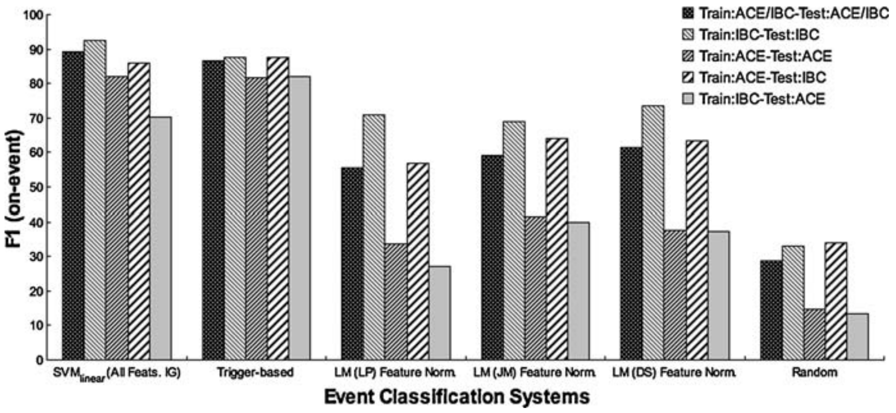


Fig. 6 % F1 for each approach using five combinations of training/test data for the *Die* event type

term probabilities are estimated in a way that is proportional to their global distribution in the training corpus. To confirm this, we evaluate each system for the *Die* event using five combinations of training/test data and compare resulting F1 scores for the *on-event* class. These results are shown in Fig. 6 where:

– **Train:ACE/IBC-Test:ACE/IBC** signifies when a mixture of data from the IBC and ACE datasets was used for both training and testing purposes during the 10-fold cross validation process.
– **Train:IBC-Test:IBC** signifies when only the IBC dataset was used for both training and testing purposes during the 10-fold cross validation process.
– **Train:ACE-Test:ACE** signifies when only the ACE dataset was used for both training and testing purposes during the 10-fold cross validation process.
– **Train:ACE-Test:IBC** signifies when the ACE data was used for training and the IBC data was used for testing.
– **Train:IBC-Test:ACE** signifies when the IBC data was used for training and the ACE data was used for testing.

The results shown in this graph suggest that the language modeling based method appear to be the most sensitive approach to changes in training source. For example, the *on-event* F1 score of all three models are reduced by approximately 40% when the *Train:IBC-Test:ACE* training/test combination is used compared to the *Train:ACE-Test:IBC* combination. This suggests that the term probabilities are less accurate when a more homogeneous dataset is used to estimate them during the training phase. The SVM scores also vary considerably across each of the difference combinations, but to a lesser extent. The results of the trigger-based classification system vary the least across the different combinations of training/test data since it is an unsupervised method and does not rely on training data to learn how to detect *on-event* sentences. Overall, using the *Train:IBC-Test:ACE* combination for training/testing produces the poorest result across all approaches for this task suggesting that when homogeneous datasets are used for training, the systems find it more difficult to correctly classify instances that contain events described across more diverse contexts, topics and circumstances. To summarise: for this task we have found that a heterogeneous training dataset produces more accurate classifiers, regardless of whether the test data is heterogeneous or homogenous.

6 Error analysis

For this classification task we investigated three different approaches for identifying sentences in a document that describe instances of a given event type, i.e., a machine learning, a language modeling and a manual knowledge engineering-based approach. In this section, we examine in detail the types of errors generated by the three approaches when the target event type is *Die*. We also analyse the amount of overlap that exists between the correct decisions produced by each system for this target type. We do this in order to determine if it would be worthwhile combining the approaches in some way with the aim of reducing the overall error rate for this task.

The errors produced by all three approaches can be classified into four broad categories. The first error type corresponds to instances, which in order to be correctly classified, require logical inference and external knowledge. We refer to this form of error as an ***inference error***. The following are typical examples for the *Die* event type: "The baby fell 80 feet" and "America does care somewhat that you've lost your leader". Humans have no problem resolving such examples. For instance, a human could infer (using world knowledge) that if a baby falls 80 feet it has very little chance of survival. However, a machine does not readily have this information available to it during the decision making process. In the second example, external knowledge or additional context is also required to infer that "lost your leader" indicates that the leader has actually died.

The second type of error identified corresponds with cases that describe continuous instances of the target event type. That is, the events are not discrete and as such have no specific location or time information attached to them. We refer to this form of error as a ***continuous error***. Examples for the *Die* event type include: "Roadside bombs account for up to 80% of U.S. deaths", "Most guns may have fallen silent but the death toll in Iraq continues to rise", and "It's like a never ending circle of violence, death and destruction". In these examples, the target event is described at the *topic* level rather than at a finer *event* level. More specifically, an exact instance of the *Die* event is not being discussed here. These false positive errors are difficult to identify since the containing sentences often exhibit the same vocabulary and features as *on-event* sentences.

In the third type of error, we have identified sentences that require deeper semantic analysis in order for the correct classification decision to be made. As such, unlike *inference errors*, no external knowledge or additional context can be used to resolve these classification errors. We refer to this form of error as a ***semantic error***. Typical examples include: "If we let this go unchecked, people will die.", "He would be the first in a really long time to actually be killed." and "They are threatening to kill the hostages.". It is clear that more complex *compositional semantic analysis* is required in order to correctly classify such cases.

The last error type identified corresponds with ***trigger-less errors***. That is, false positive cases that typically contain terms that commonly occur in *on-event* sentences (e.g., 'hospital', 'wounded', 'detained'), but are not accompanied with the appropriate target event trigger term. Typical examples include: "Two wounded women were taken to hospitals in Baghdad and Ramadi.", and "On Sunday night U S troops detained ex army Gen Mumtaz al Taji at a house in Baquba about 30 miles north of Baghdad."

Figure 7 depicts the percentage breakdown of the four major error types produced by the unigram language model smoothed using *Jelinek-Mercer* (left), the SVM (middle) and the trigger-based classifier (right). After studying these pie charts we notice the following trends:

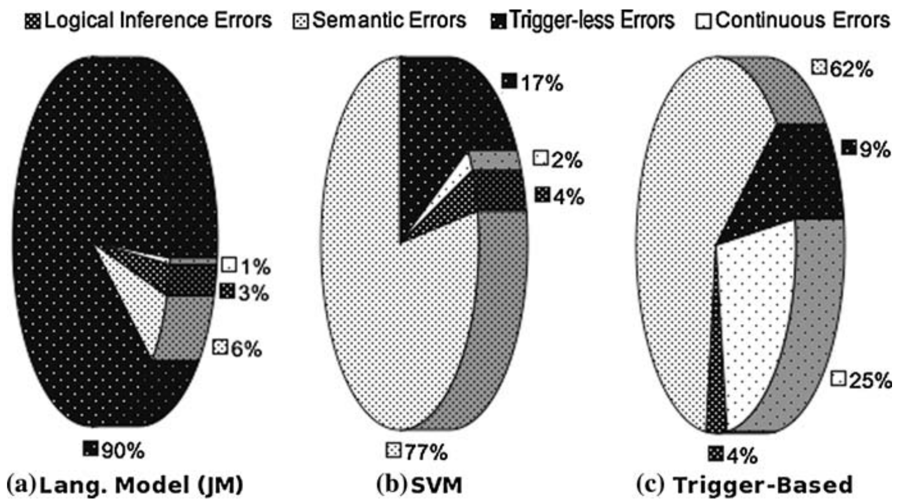


Fig. 7 Percentage breakdown of the four major error types in each event classification system where the target event is *Die*

- 90% of the errors produced by the language model are *trigger-less* errors where the sentences do not describe instances of the target event type, but were classified as *on-event* instances.
- 77% of the errors produced by the SVM are *semantic* errors. These are mainly cases that report instances of the target event type, but have been misclassified as *off-event* sentences by the SVM because the "true" semantic meaning of the sentence was not correctly established.
- 17% of the errors produced by the SVM are *trigger-less* errors. These are false positive sentences that do not contain any trigger terms, but do contain terms such as 'weapon', 'gun' and 'war which are often found in *on-event Die* sentences. Both the language modeling and SVM approaches are using contextual terms surrounding trigger words as discriminative features in their classification decisions. However, the occurrence of a strong contextual term can result in a classification error when the accompanying trigger term does not occur. This is a problem with learning methods where term independence is assumed.
- 62% of the errors produced by the trigger-based classifier are *semantic* errors. When we examined these errors closely we found that 50% were false positives and the remaining 50% were false negatives. Interestingly, many of the false positives errors consisted of sentences that contained the term 'suicide' as part of a noun phrase such as "suicide attack" or "suicide car bomber". Also, we found that the false negative cases were mainly sentences such as "The police found the bodies of seven men in various parts of Baghdad." and "Three more bodies were found in the New Baghdad district where human bodies were found dead".
- 25% of the errors produced by the trigger-based classifier are *continuous* errors. These sentences talk about the target event in the general sense, but no specific instance of it is reported.
- 9% of the errors produced by the trigger-based classifier are *trigger-less* errors. These are mainly cases where the sense of the trigger term is not connected with *Die*. For example, the term 'execute' has many other distinct meanings associated with it other

than ‘to be executed’. Examples include sentences such as “She **executed** the speech perfectly with no problems.”. Also, when the term ‘executive’ is stemmed it becomes ‘execute’ which explains why sentences such as “Reuters chief **executive** Tom Glocer called for a comprehensive investigation into this event.” were incorrectly classified as *on-event* sentences by the trigger-based classifier.

One recurring observation from this error analysis is that the tense of the sentence and the part of speech of its trigger term(s) tend to act as good indicators for this classification task. For example, when a trigger term occurs as a past participle (“was killed”) it usually indicates that the event has already occurred. Whereas if it occurs as a present participle (“is killing”) or a noun (“the dead”), this is not necessarily the case. In the existing SVM presented in Sect. 4.1, we only capture the presence or absence of each POS tag as features. However, it appears from this error analysis that these features alone are not powerful enough to convey the underlying semantics of the sentence. Also, they tell us nothing about the grammatical characteristics of the terms acting on the specific key terms such as the trigger term(s). To combat this problem, we add the following additional feature set to the SVM:

– **Combined Term and Lexical Features:** For each stemmed term in the corpus we create a feature for it and concatenate it with its part of speech in this context. For example, the stem **“kil”** can occur as a past tense verb (“The device **kil**led seven people”), as a present participle (“is **kil**ling thousands”), as a past participle (“Seven were **kil**led ”), as a noun (“the **kil**ling”) or as a plural noun (“the targeted **kil**lings”). So for this stem we add five additional features: {*kil_VBD, kil_VBG, kil_VBN, kil_NN, kil_NNS*}.

The resulting overall classification accuracy as well as the Precision, Recall and F1 for each class of the modified SVM (SVM_{linear} (*Modified*)) and original SVM (SVM_{linear} (*AllFeats. IG*)) is shown in Table 6. We also include the results of the existing trigger term based system for comparison purposes. From these results we see a minimal increase in overall performance as a result of adding these additional features. Although the increase is not huge it is encouraging to see that the modified classifier correctly classifies previous errors such as “Unexploded ordnance UXO in northern Iraq are killing and maiming dozens of people every day” where the trigger term (‘killing’) occurs as a present participle. One of the reasons for this is that the *kill_VBG* feature, which captures the fact that the verb ‘to kill occurs as a present participle, is ranked as the 8th most discriminative feature used by the modified SVM.

The last question we wish to address in this section is to determine whether it is worthwhile combining the output of these systems (using for example, an ensemble

Table 6 % Precision, Recall and F1 of the *on-event* class achieved by the SVM with new features where the target event type is *Die*

Algorithm	On-event class		
	Precision	Recall	F1
SVM_{linear} (Modified)	93.19	92.27	92.72
SVM_{linear} (All Feats. IG)	91.83	92.94	92.38
Trigger-based	83.34	92.51	87.66

These scores are generated from the IBC dataset using 10-fold cross validation

Table 7 % *on-event* overlap between the event classification systems

$OnEvent_{overlap}$	SVM_{linear} (Modified)	LangModel (JM)	Trigger-based
SVM_{linear} (Modified)	100%	–	–
LangModel (JM)	79.8%	100%	–
Trigger-based	94.7%	81.3%	100%

Table 8 % *off-event* overlap between the event classification systems

$OffEvent_{overlap}$	SVM_{linear} (Modified)	LangModel (JM)	Trigger-based
SVM_{linear} (Modified)	100%	–	–
LangModel (JM)	13.86%	100%	–
Trigger-based	36.62%	13.04%	100%

method) in an attempt to reduce the overall error rate of the task. One way of answering this question is to determine whether or not each system is classifying the same instances correctly, since high performing yet diverse systems can increase performance when combined (Ng and Kantor 2000). Table 7 shows the percentage of correct classifications generated by each system pair for the *on-event* class. We can see that the $OnEvent_{overlap}$ between the system pairs ranges between 79.8% and 94.7%. The most similar system pair is the SVM and trigger-based classifier. More specifically, these systems correctly classify almost 95% of the same *on-event* sentences. In contrast, for $OffEvent_{overlap}$ scores (as shown in Table 8), we see that systems are tending to produce different off-event errors. This implies that a combination experiment could be beneficial if a voting type scheme were employed where contradicting classifications were resolved by counting votes across systems. However, the overall gains will probably be minimal. We leave this combination experiment for future work.

7 Conclusions and future directions

Sentence level event classification is an important first step for many NLP applications such as QA and summarisation systems. For each event type used in our experiments we compared a variety of approaches for identifying sentences that described instances of that type.

Overall we have shown that the most effective approach for this task depends on the event type being detected. For broad vocabulary event types such as *Attack* and *Transport*, the SVM appears to be the most appropriate approach. For more specific types such as *Charge-Indict* and *Injure*, the trigger-based classification system proved to be the most powerful approach. In general, the language models were the least effective out of the three systems across all event types and datasets. We also observed that the performance of each approach for the *on-event* class varies considerably across the event types. Also, we found that terms alone prove to be the most powerful feature set used by the SVM for this classification task. Additional features such as the presence/absence of the *VBD* part of speech tag and sentence position can also act as good indicators regardless of the event type in question. Other features such as the presence/absence of a numerical token was found to be useful for certain types such as the *Die* event. Finally, our experiments also revealed that the type of dataset used for training significantly affects the performance of the supervised approaches. Specifically, heterogeneous datasets with a rich vocabulary proved to be more suitable for training purposes and thus produced better performing classifiers on this task.

As part of our future work, we intend to focus on an issue that was not discussed in this paper, namely, *event co-reference resolution*. In a real-world event detection application, such as calculating an Iraq war body count from news articles, the system should also be capable of identifying multiple instances of the same event to ensure that a particular *Die* event is not counted multiple times. We are currently investigating the applicability of clustering techniques as a means of identifying co-referring event sentences within and between distinct news articles reporting on the same event.

Acknowledgements This research was supported by the Irish Research Council for Science, Engineering & Technology (IRCSET) and IBM under grant RS/2004/IBM/1. The authors wish to thank the members of the Language Technology Research Group at the University of Melbourne, NICTA for their helpful discussions regarding this research. We also wish to extend our gratitude to Anthony Brew and Prof. Pádraig Cunningham from the Machine Learning Research Group at University College Dublin for their advice regarding some of the machine learning aspects of this research.

References

Aery, M., & Chakravarthy, S. (2004). emailsift: Mining-based approaches to email classification. In *Proceedings of the 27th international ACM SIGIR conference on research and development in information retrieval* (pp. 580–581). New York: ACM.

Ahn, D. (2006). The stages of event extraction. In *Proceedings of the ACL workshop on annotating and reasoning about time and events* (pp. 1–8). Morristown, NJ: Association for Computational Linguistics.

Allan, J., Carbonell, J., Doddington, G., Yamron, J., & Yang, Y. (1998). Topic detection and tracking pilot study. Final report. In *Proceedings of DARPA broadcast Nem transcription and understanding workshop* (pp. 194–218).

Allan, J., Gupta, R., & Khandelwal, V. (2001). Temporal summaries of news topics. In *Proceedings of the 24th international ACM SIGIR conference on research and development in information retrieval* (pp. 10–18). New York: ACM.

Atkinson, M., Piskorski, J., Pouliquen, B., Steinberger, R., Tanev, H., & Zavarella, V. (2008). Online-monitoring of security-related events. In *Proceedings of the 22nd international conference on computational linguistics* (pp. 1–4). Manchester, UK: Coling 2008 Organizing Committee.

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

Cunningham, P. (2008). Dimension reduction. In M. Cord & P. Cunningham (Eds.), *Machine learning techniques for multimedia* (pp. 91–112). Berlin: Springer.

Curran, J., Clark, S., & Bos, J. (2007). Linguistically motivated large-scale NLP with C&C and Boxer. In *Proceedings of the 45th annual meeting of ACL, Demonstrations Session* (pp. 29–32).

Daniel, N., Radev, D., & Allison, T. (2003). Sub-event based multi-document summarization. In *Proceedings of the HLT-NAACL 03 workshop on text summarization* (pp. 9–16). Morristown, NJ: Association for Computational Linguistics.

Dredze, M., Lau, T., & Kushmerick, N. (2006). Automatically classifying emails into activities. In *Proceedings of the 11th international conference on intelligent user interfaces* (pp. 70–77). New York: ACM.

Filatova, E., & Hatzivassiloglou, V. (2004). Event-based extractive summarization. In *Proceedings of the ACL workshop on summarization* (pp. 104–111).

Grishman, R., Huttunen, S., & Yangarber, R. (2002). Real-time event extraction for infectious disease outbreaks. In *Proceedings of the HLT'02 conference* (pp. 366–369).

Hripcsak, G., Bakken, S., Stetson, P., & Patel, V. (2003). Mining complex clinical data for patient safety research: A framework for event discovery. *Journal of Biomedical Informatics, 36*(1/2), 120–130.

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2000). *A practical guide to support vector classification*. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.6.3096.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In C. Nédellec & C. Rouveirol (Eds.), *Proceedings of the 10th European conference on machine learning* (pp. 137–142.). Springer, Heidelberg, DE, Chemnitz, DE.

Keerthi, S. S., & Lin, C.-J. (2003). Asymptotic behaviors of support vector machines with gaussian kernel. *Neural Computation, 15*(7), 1667–1689.

King, G., & Lowe, W. (2003). An automated information extraction tool for international conflict data with performance as good as human coders: A rare events evaluation design. *International Organization, 57*(03), 617–642.

Kraaij, W., & Spitters, M. (2003). Language models for topic tracking. In B. Croft & J. Lafferty (Eds.), *Language models for information retrieval*. Norwell, MA: Kluwer Academic Publishers.

Larkey, L., & Croft, B. (1996). Combining classifiers in text categorization. In *Proceedings of the 19th international ACM SIGIR conference on research and development in information retrieval* (pp. 289–297). New York: ACM.

Lewis, D. (1992). An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th international ACM SIGIR conference on research and development in information retrieval* (pp. 37–50). New York: ACM.

Li, W., Wu, M., Lu, Q., Xu, W., & Yuan, C. (2006). Extractive summarization using inter and intra event relevance. In *Proceedings of the 44th annual meeting of ACL* (pp. 369–376). Morristown, NJ: Association for Computational Linguistics.

Manning, C., & Schütze, H. (1999). *Foundations of statistical natural language processing*. Cambridge, MA: MIT Press.

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive bayes text classification. In *Proceedings of the AAAI-98 Workshop on Learning for Text Categorization* (pp. 41–48). AAAI Press.

Miller, G. (1995). Word Net: A lexical database for english. *Communications of the ACM, 38*(11), 39–41.

Moschitti, A., Quarteroni, S., Basili, R., & Manandhar, S. (2007). Exploiting syntactic and shallow semantic kernels for question answer classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics* (pp. 776–783). Association for Computational Linguistics.

Murff, H., Patel, V., Hripcsak, G., & Bates, D. (2003). Detecting adverse events for patient safety research: A review of current methodologies. *Journal of Biomedical Informatics, 36*(1/2), 131–143.

Naughton, M., Stokes, N., & Carthy, J. (2008). Investigating statistical techniques for sentence-level event classification. In *Proceedings of the 22nd international conference on computational linguistics* (pp. 617–624). Manchester, UK: Coling 2008 Organizing Committee.

Ng, K. B., & Kantor, P. B. (2000). Predicting the effectiveness of naïve data fusion on the basis of system characteristics. *Journal of the American Society for Information Science (JASIS), 51*(13), 1177–1189.

Platt, J. C. (1999). Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, & A. J. Smola (Eds.), *Advances in Kernel methods: Support vector learning* (pp. 185–208). Cambridge, MA: MIT Press. ISBN:0-262-19416-3.

Porter, M. F. (1997). An algorithm for suffix stripping. In K. Sparck Jones & P. Willett (Eds.), *Readings in information retrieval* (pp. 313–316). San Francisco, CA: Morgan Kaufmann Publishers Inc. ISBN:1-55860-454-5.

Quinlan, R. (1993). *C4.5: Programs for machine learning*. San Francisco, CA: Morgan Kaufmann Publishers Inc.

Sahami, M. (1996). Learning limited dependence Bayesian classifiers. In *Proceedings of the 2nd international conference on knowledge discovery and data mining* (pp. 335–338). AAAI Press.

Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998). A Bayesian approach to filtering junk e-mail. In *Proceedings of the AAAI-98* (pp. 1–8).

Saurí, R., Knippen, R., Verhagen, M., & Pustejovsky, J. (2005). Evita: A robust event recognizer for QA systems. In *Proceedings of the conference on HLT and empirical methods in natural language processing* (pp. 700–707).

Schultz, J., & Liberman, M. (1999). Topic detection and tracking using idf weighted cosine coefficient. In *Proceedings of the DARPA broadcast news workshop* (pp. 189–192). Morgan Kaufmann Publishers Inc.

Schwartz, R., Imai, T., Nguyen, L., & Makhoul, J. (1997). A maximum likelihood model for topic classification. In *Proceedings of Eurospeech* (pp. 1455–1458).

Segal, R., Crawford, J., Kephart, J., & Leiba, B. (2004). Spamguru: An enterprise anti-spam filtering system. In *Proceedings of the first conference on email and anti-spam.*

Segal, R., & Kephart, J. (2000). Incremental learning in swift file. In *Proceedings of the 17th international conference on machine learning* (pp. 863–870). Morgan Kaufmann Publishers Inc.

Surdeanu, M., Ciaramita, M., & Zaragoza, H. (2008). Learning to rank answers on large online qa collections. In *Proceedings of the 46th annual meeting of the Association of Computational Linguistics; Human Language Technologies (ACL-HLT)*. Association for Computational Linguistics. http://grupoweb.upf.es/hugoz/pdf/mihai_acl08.pdf

Tang, L., & Liu, H. (2005). Bias analysis in text classification for highly skewed data. In *Proceedings of the 5th IEEE international conference on data mining* (pp. 781–784). Washington, DC: IEEE Computer Society.

Vanderwende, L., Banko, M., & Menezes, A. (2004). Event-centric summary generation. In *Working notes of DUC 2004* (pp. 76–81).

Walker, C., Strassel, S., Medero, J., & Consortium, L. D. (2006). *ACE 2005 multilingual training corpus*. Linguistic Data Consortium, University of Pennsylvania.

Walls, F., Jin, H., Sista, S., & Schwartz, R. (1999). Topic detection in broadcast news. In *Proceedings of the DARPA broadcast news workshop* (pp. 193–198). Morgan Kaufmann Publishers Inc.

Witten, I., & Frank, E. (2000). *Data mining: Practical machine learning tools and techniques with Java implementations*. San Francisco, CA: Morgan Kaufmann Publishers Inc.

Wu, M. (2006). Investigations on event-based summarization. In *Proceedings of the COLING/ACL student research workshop* (pp. 37–42). Morristown, NJ: Association for Computational Linguistics.

Yamron, J. P., Gillick, L., van Mulbregt, P., & Knecht, S. (2002). Statistical models of topical content. In *Topic detection and tracking: Event-based information organization* (pp. 115–134). Norwell, MA: Kluwer Academic Publishers. ISBN:0-7923-7664-1.

Yang, Y., Ault, T., Pierce, T., & Lattimer, C. (2000). Improving text categorization methods for event tracking. In *Proceedings of the 23rd international ACM SIGIR conference on research and development in information retrieval* (pp. 65–72). New York: ACM.

Yang, Y., Carbonell, J., Brown, R., Pierce, T., Archibald, B., & Liu, X. (1999). Learning approaches for detecting and tracking news events. *IEEE Intelligent Systems, 14*(4), 32–43.

Yang, Y., Pierce, T., & Carbonell, J. (1998). A study of retrospective and on-line event detection. In *Proceedings of the 21st international ACM SIGIR conference on research and development in information retrieval* (pp. 28–36). New York: ACM.

Zhai, C., & Lafferty, J. (2001). A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th international ACM SIGIR conference on research and development in information retrieval* (pp. 334–342). New York: ACM.