Sentence-Level Event Classification in Unstructured Texts

Martina Naughton, Nicola Stokes, and Joe Carthy

School of Computer Science & Informatics, University College Dublin, Ireland
{martina.naughton, nicola.stokes, joe.carthy}@ucd.ie

Technical Report UCD-CSI-2008-07
September 2008

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. We compare the performance of two approaches to this task: a Support Vector Machine (SVM) classifier and a Language Modeling (LM) approach. We also investigate a rule-based method that uses hand-crafted lists of 'trigger' terms derived from WordNet. We use two datasets in our experiments and test each approach using six different event types, i.e, *Die*, *Attack*, *Injure*, *Meet*, *Transport* and *Charge-Indict*. Our experimental results indicate that although the trained SVM classifier consistently outperforms the language modeling approach, our rule-based system marginally outperforms the trained SVM classifier on three of our six event types. We also observe that overall performance is greatly affected by the type of corpus used to train the algorithms. Specifically, we have found that a homogeneous training corpus that contains many instances of a specific event type (i.e., *Die* events in the recent Iraqi war) produces a poorer performing classifier than one trained on a heterogeneous dataset containing more diverse instances of the event (i.e., *Die* events in many different settings, for example, traffic accidents, natural disasters etc.). Our heterogeneous dataset is provided by the ACE (Automatic Content Extraction) initiative, while our novel homogeneous dataset consists of news articles and annotated *Die* events from the Iraq Body Count (IBC) database. Overall, our results show that the techniques presented here are effective solutions to the event classification task described in this paper, where F1 scores of over 90% are achieved.

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 situations in the world (1). 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 (2; 3; 4; 5; 6). 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. More specifically, 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 statistical methods for identifying the sentences in a document that describe one or more instance of a specified event type. We treat this task as a text classification problem where each sentence is either classified as one that contains an instance of the target event or 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. One advantage of language modeling for text classification is that instead of explicitly pre-computing features and selecting a subset based on arbitrary decisions (as is often the case with standard classification learning approaches), the language modeling approach considers all terms occurring in the text as candidate features, and implicitly estimates the contribution of each feature in the final model. We estimate a series of uni-gram 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' as positive instances may catch many positive instances. Nevertheless, there are instances where this

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

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 hand crafted list of 'trigger' terms found in WordNet (7). This system is compared with the SVM and uni-gram 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 (8) that was annotated for 33 different event types. However, 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 Iraq Body Count (IBC) database[2] manually annotated with *Die* event type. This dataset arose from a larger research 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 type.

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 performance of our supervised approaches on this classification task are significantly affected by the type of dataset used for training. 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, we investigate the applicability of Machine Learning and Language Modeling based approaches for Event Classification at a sentence-level. This work is novel because related research efforts have focused on event detection at either the term/phrasal level or the document level, rather than at the sentence level; a granularity which is favored by many IR, QA and Summarisation applications. We also introduce a new event dataset of Iraqi war articles where all instances of the *Die* event type have been manually annotated at the sentence level.

The remainder of this paper is organised as follows. We begin with a brief description of background research and related work in Section 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 Section 5. The common errors

[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 contract, 'specific' event types can be expressed using a more limited set of vocabulary terms. These concepts are described in more detail in 5.2.

produced by each approach are analysed in Section 6. Finally, in Section 7, we conclude with a discussion of our experimental observations and our intentions for future work [4].

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 ten years. Many event extraction technology have been reported in the literature. For example, event extraction systems capable of detecting disease outbreaks (10), conflict events (11; 12), and natural disaster events in a multilingual setting (12). Moreover, the NIST sponsored Topic Detection and Tracking (TDT) project, which 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 (13; 14; 15). 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 (16; 17).

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 occurence involving zero or more ACE entities[5], 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 events 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 the 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 (18), 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.

[4] Some initial experiments by the authors on this work are published in (9). However, substantially more experimental results and analysis are presented in this publication.

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

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 (14; 19), 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 Neighbor (kNN) classification, Decision Tree induction and a variety of LM approaches (14; 13; 15; 20; 21; 22; 23). However, these TDT tasks were somewhat restrictive in the sense that detection is carried out at document level. Our work differs from TDT research since event detection 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 (24; 25) and automatic foldering (26; 27; 28) systems) have employed text classification techniques such as naïve Bayes, rule learners, and SVMs.

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 with 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 (29). 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.

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**
# Docs.	154	50	235	84	181	43	332
Avg. Doc. Length	29.2	29.7	29.6	31.4	32.8	14.8	25.9
Avg. Ev. per Doc.	2.31	1.64	3.55	1.55	2.55	1.72	4.6
Total # Events	357	82	836	131	462	74	1528

When the IBC and ACE datasets are compared, we find that there are some 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 Figure 1 for specific examples). This greater diversity in vocabulary within topics is also clear when we analyse and 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 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.

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 datsets.

4 Event Detection as Classification

In this paper, we treat the event detection as a sentence level binary classification task. As such, for a given target event type, each sentence is assigned 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 Support Vector Machines (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 (30). In our experiments we used a relatively efficient implementation of an SVM called the Sequential Minimal Optimisation (SMO) algorithm (31) which is provided through the Weka framework (32).

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 (33)), with a frequency in the training data greater than two, were used as term features. All stopwords were removed from this feature set.
- **Noun Chunks:** All noun chunks with a frequency greater than two in the training data were also used as a feature. Examples include 'American soldier', 'suicide bomb'.
- **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, available with the C&C Toolkit (34). 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*, *VP*, *PP*, *ADJP*, *ADVP* and so on.
- **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. Timestamps were identified using in-house software developed by members of the Language Technology research group at the University Melbourne[6].

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

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 half.

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 $c \in C = \{C_1 \dots C_{|C|}\}$. One way to do this is to choose the category that has the largest posterior probability given the training text:

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

Specifically, we construct a language model $LM(c_i)$ for each class c_i. All models built are unigram models that use a maximum likelihood estimator to approximate term probabilities. According to this model (built from sentences $\{s_1 \dots s_m\}$ belonging to class c_i in the training data) we can calculate the probability that term w was generated from class c_i as:

$$P(w|LM(c_i)) = \frac{tf(w, c_i)}{|c_i|} \tag{2}$$

where $tf(w, c_i)$ is the term frequency of term w in c_i (that is, $\{s_1 \dots s_m\}$) and $|c_i|$ is the total number of terms in class c_i. We make the usual assumptions that word co-occurences are independent. As a result, the probability of a sentence is the product of the probabilities of its terms. We calculate the probability that a given test sentence s_k belongs to class c_i as follows:

$$P(s_k|LM(c_i)) = \prod_{w \in s_k} P(w|LM(c_i)) \tag{3}$$

However, this model will 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 (35; 36). For this reason, we experiment with various smoothing techniques and compare their effects on classification performance in Section 5.

One of the simplest proposed solutions to this problem is the *LaPlace* smoothing method (37). Similar to (38) we use a variant of this technique by adding 0.01 to the numerator and multiply the denominator by 1.01 as follows:

$$P_{lp}(w|LM(c_i)) = \frac{tf(w, c_i) + 0.01}{|c_i| \cdot 1.01} \tag{4}$$

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 co-efficient λ to control the influence of each model as follows:

$$P_{jm}(w|LM(c_i)) = (1-\lambda)P(w|LM(c_i)) + \lambda P(w|C) \tag{5}$$

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 $\lambda \in [0, 1]$, to find an 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:

$$P_{ad}(w|LM(c_i)) = \frac{max(tf(w, c_i) - \delta, 0) + \delta|c_i|_u \cdot P(w|C)}{|c_i|} \tag{6}$$

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 $\delta \in [0, 1]$.

For this classification task, we normalise all numeric references, locations, person names and organisations to *DIGIT*, *LOC*, *PER*, and *ORG* respectively. This helps to reduce the dimensionality of our models, and improve their classification accuracy, particular in cases where unseen instances of these entities occur in the test data.

4.3 Trigger Based Event Classification

According to the ACE annotation guidelines[7] event instances are identified in the text by finding event triggers that explicitly mark the occurrence of the

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

event. As a result, each event instance tagged in our datasets have 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 the *Die* event type. Therefore, the trigger based event classification system will select sentences containing one or more candidate trigger term as positive *on-event* instances. 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[8].

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

4.4 Baseline Systems

We compare the performance of the trained SVM, uni-gram language models and the trigger based classification system against the following plausible baseline systems:

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

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

5 Evaluation Methodology & 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

[8] Trigger term lists are available for download at:
`http://inismor.ucd.ie/~martina/triggerLists.html`

our datasets where approximately 90% of the instances belong to the *off-event* class) this measure can be misleading. Instead we report *on-event* and *off-event* evaluation scores 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:

$$Precision_s = \frac{a}{b}, \quad Recall_s = \frac{a}{c} \quad and$$

$$F1_s = \frac{2 \times Precision_s \times Recall_s}{Precision_s + Recall_s} \tag{7}$$

Precision, Recall and F1 scores for the *off-event* class are formulated in a similar fashion. In our evaluation, we report each of these measures for both classes as well as the overall classification accuracy score of each system. For completeness, classification accuracy is also calculated as follows: if r is the total number of instances correctly classified (true positives and true negatives) by system s, and n is the total number of human annotated instances in the collection, then the classification accuracy of s can be formulated as:

$$Accuracy_s = \frac{r}{n} \tag{8}$$

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. How does the performance of each approach differ across the six event types explored in this paper?
2. Do additional linguistic features improve the classification performance of the SVM?
3. How effective are the uni-gram language models as event classifiers?
4. What is the effect of varying the size of classifier training data?
5. What is the effect of homogeneous versus heterogeneous training data on classification performance?

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 3 shows the Precision, Recall and F1 scores achieved for the *on-event* and *off-event* classes as well as the overall classification accuracy obtained by each approach. Two variations of the SVM were built. The first version (denoted in the table by *SVM(All Feats. IG)*) was trained using all the features

Table 3. % Precision, Recall and F1 for both classes as well as the classification accuracy achieved by all algorithms where the target event type is *Die*. These scores are generated from the IBC dataset using 50:50 random train/test splits averaged over 5 runs.

Algorithm	***on-event* class**			***off-event* class**			**acc**
	prec	**rec**	**f1**	**prec**	**rec**	**f1**	
SVM(All Feats. IG)	**90.61**	**89.87**	**90.23**	**96.15**	**97.26**	**96.70**	**94.60**
SVM(All Feats.)	89.63	88.52	89.06	96.08	96.49	96.28	94.45
Trigger based	83.10	93.34	87.92	97.25	93.24	95.20	93.09
LangModel(DS)	63.11	82.4	71.46	93.13	83.16	87.86	82.98
LangModel(JM)	59.46	86.01	70.31	94.22	79.53	86.25	81.22
LangModel(LP)	59.22	79.56	67.89	91.89	80.88	86.03	80.54
Majority Class	0.0	0.0	0.0	74.50	100.00	85.38	74.17
Random	26.57	51.73	35.10	75.58	50.0	60.18	50.34

(approximately 5000) to encode each training/test instance where the features were reduced using Information Gain (IG). In the second version, the same set of features were used, but no feature reduction was carried out (denoted in the table by *SVM(All Feats.)*). *LangModel(JM)*, *LangModel(DS)* and *LangModel(LP)* represent uni-gram language models smoothed using *Jelinek-Mercer*, *Discount Smoothing* and *LaPlace* techniques, respectively.

Overall, results in this table suggests that the SVM using Information Gain(IG) for feature selection purposes is the most effective method for correctly classifying both *on-event* and *off-event* sentences using this data for this event type. When IG is not used we see a marginal decrease of approximately 1% in the F1 score achieved by both classes. The fact that each version of the SVM obtains an *on-event* F1 score of about 90% is extremely encouraging when you consider the large skew in class distribution that is present here (i.e., the majority of training instances belong to the *off-event* class). The results in this table also indicate that the trigger based classification baseline approach performs very well, achieving similar scores to both versions of the SVM. This 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 7% 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. In general, the language modeling based techniques are not as effective as the SVM approach or trigger based system for this classification task. Nevertheless, from Table 3 we see that all language models achieve an F1 score of approximately 70% for the *on-event* class and approximately 86% for the *off-event* class when only half of the IBC data is used to build the models.

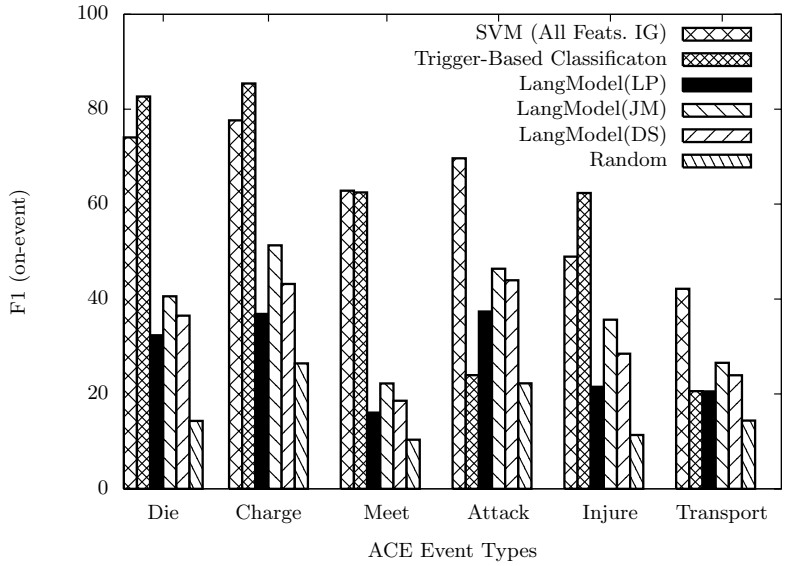

Fig. 2. % F1 of the *on-event* class achieved by each method for all six event types. These scores are generated from the ACE dataset using 50:50 random train/test splits averaged over 5 runs.

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 by all approaches on the six event types defined in the ACE data. Overall, these results indicate that the performance of each approach for the *on-event* class varies considerably across the event types. For example, the trigger based system obtains an F1 score of approximately 60% for the *Meet* and *Injure* types. However, there is almost a 40% difference in F1 when the target event is *Attack* or *Transport*. Also, when we examine how the SVMs performance varies across event types, we see that for the *Charge-Indict* event an F1 score of approximately 77% is achieved; however, when the target type is *Injure* the F1 score is approximately 48%.

The results in Figure 2 also show that each approach tends to marginally perform better on the *Charge-Indict* event type. One plausible reason for this is that this event is very 'specific' where only a few terms (e.g., 'charge', 'accuse', 'indict', 'incriminate') are typically associated with its occurrence. 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*. 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 Figure 3. We see from Figure 2 that the SVM performs well on such event types, outperforming the trigger based system on the the *Attack* and *Transport* events by a percentage difference of approximately 50% and 20%, respectively.

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.

In summary, both sets of results (ACE and IBC) demonstrate that the SVM and trigger based classification system are our top performing systems overall for this task.

Effectiveness of Linguistic Features To access the effectiveness of each feature set on overall classification performance, we evaluated the SVM using different feature set combinations for all event types using both datasets. These results are shown in Figure 4 where *terms*, *nc*, *lex*, *additional* denote the terms, noun chunks, lexical, and 'additional' features, respectively. See Section 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 Figure 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; however, again this increase is not statistically significant. 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

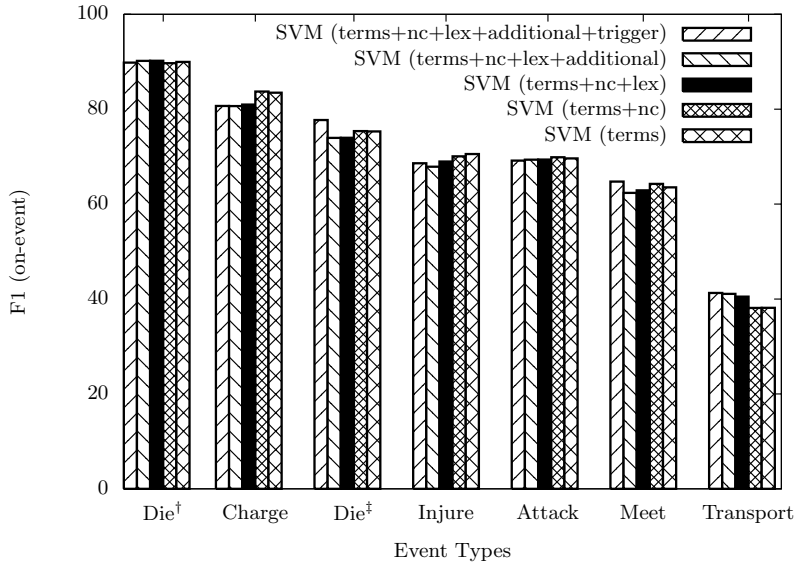

Fig. 4. % F1 for the *on-event* class achieved by the SVM using different combinations of features for the six event types where Die† and Die‡ represents the results of the *Die* event using the IBC and ACE datasets respectively. These scores are generated using 50:50 random train/test splits averaged over 5 runs.

speech tag, sentence length and sentence position are ranked as the 2^{nd}, 4^{th}, 6^{th} and 14^{th} most discriminative features respectively when the target event is *Die*. This indicates that some of the linguistic features added to the encoding process are considered by the SVM to be useful.

Effects of Varying Training Data Size The graphs in Figure 5 depict the F1 scores of both classes achieved by all methods using varying levels of training data. These graphs show that the SVM obtains an F1 score of approximately 83% for the *on-event* class and over 90% for the *off-event* class when only 10% of the data is used for training. Similarly, the language models presented obtain F1 scores of approximately 60% and 80% for the *on-event* and *off-event* classes, respectively, when only 10% of the data is used for training. All results increase gradually as the amount of training data increases. For levels of training data greater than 40% the SVM achieves marginally higher F1 scores for both classes than the other methods do.

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*

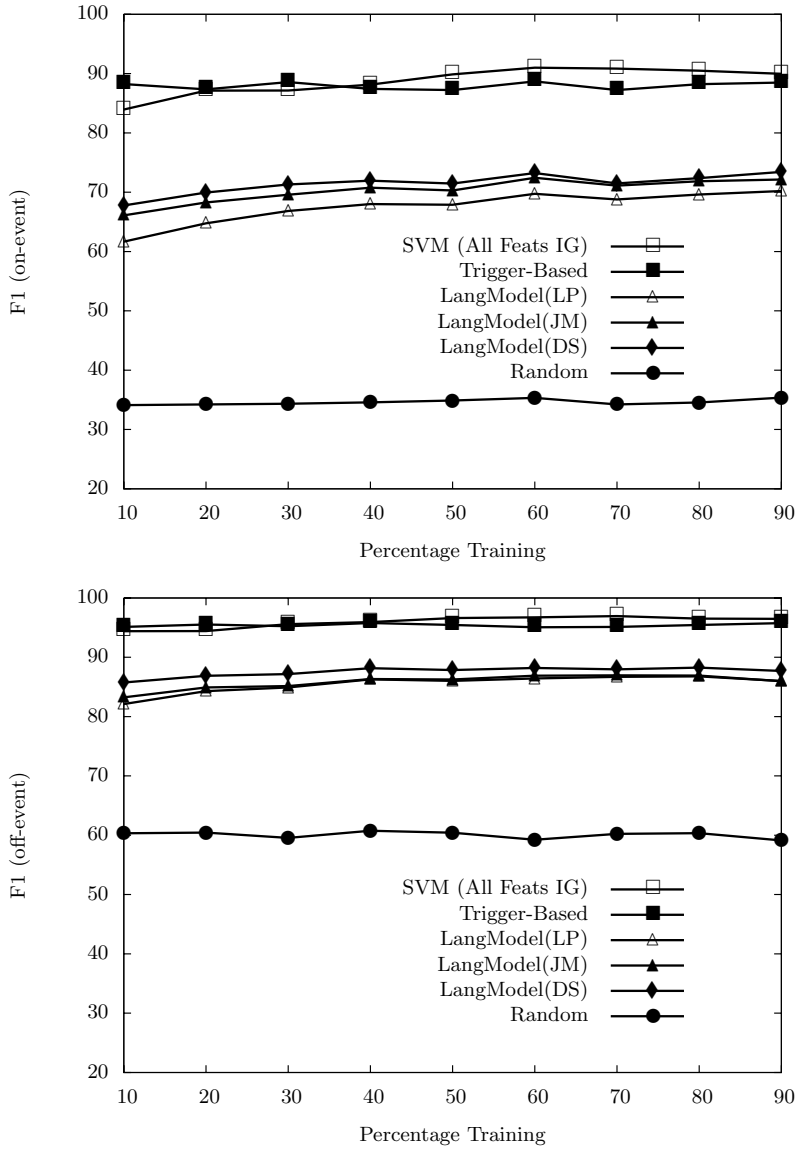

Fig. 5. % F1 for the *on-event* (top) and *off-event* (bottom) class for all methods using varying levels of training data where the target event is *Die*. These scores are averaged over 5 runs and generated using the IBC dataset.

method tend to have the least impact out of the three smoothing techniques investigated. This is due to the 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.

Homogeneous versus Heterogeneous Training Data When we compared the IBC and ACE datasets in Section 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 model's since 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 Figure 6 where:

- **Train:ACE/IBC-Test:ACE/IBC** signifies when a random 50:50 mix of both IBC and ACE data was used for training and testing.
- **Train:IBC-Test:IBC** signifies when a random 50:50 mix of IBC data was used for training and testing.
- **Train:ACE-Test:ACE** signifies when a random 50:50 mix of ACE data was used for training and testing.
- **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 hetrogeneous or homogenous.

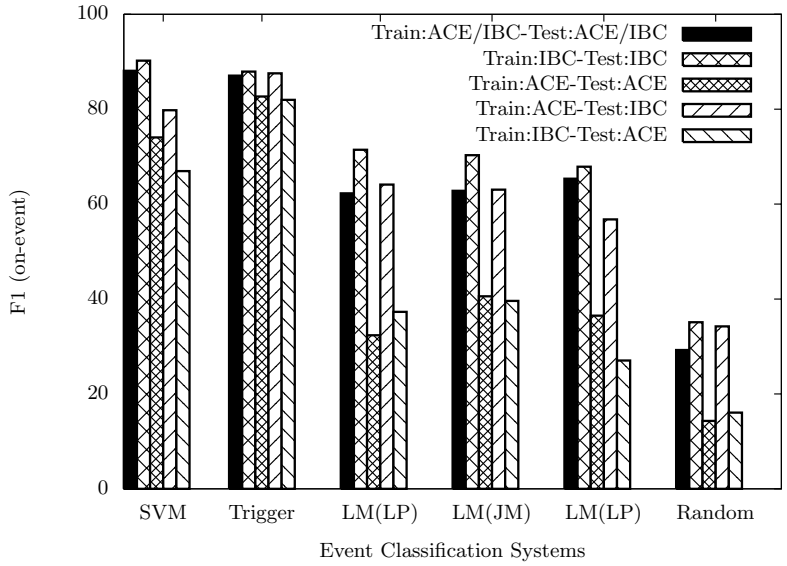

Fig. 6. % F1 for each approach using five combinations of training/test data for the *Die* event type. These scores are generated using 50:50 random train/test splits averaged over 5 runs.

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 percent 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 instances are hard 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 requires 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 completely off-topic event instances, that is an ***off-topic error***. 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.". These cases typically contain terms that

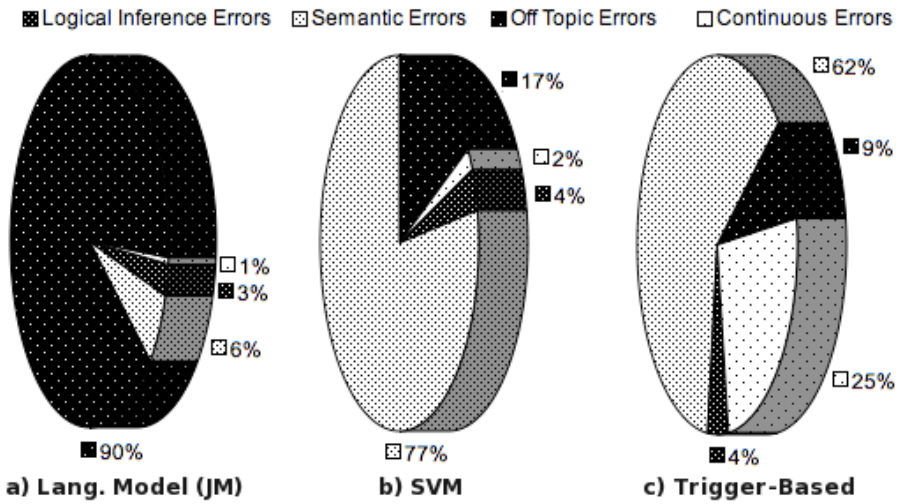

Fig. 7. Percentage breakdown of the four major error types in each event classification system where the target event is *Die*.

commonly occur in *on-event* sentences (e.g., 'hospital', 'wounded', 'detained'), but are not accompanied with the appropriate trigger term.

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:

- 90% of the errors produced by the language model are *off-topic* 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. Specifically, these are mainly cases that report the target event type, but have been missed by the SVM because the "true" semantic meaning of the sentence was not correctly established.
- 17% of the errors produced by the SVM are *off-topic* errors. These are false positives that often contain terms such as 'weapon', 'gun' and 'war which are often found in *on-event Die* event 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, 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 *off-topic* 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 the fullest and comprehensive investigation into this event." were incorrectly classified as *on-event* sentences by the trigger based classifier.

One obvious observation from this error analysis, is that the tense the sentence is written in 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 Section 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 (Modified)) and original SVM (SVM (All Feats. IG)) is shown in Table 4. We also include the results of the existing trigger term based system for comparison purposes. From these results we see a minor 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 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(39). Table 5 shows the percentage of

Table 4. % Precision, Recall and F1 for both classes as well as the classification accuracy achieved by the SVM with new features using a 50:50 training/test split where the target event type is *Die*.

Algorithm	***on-event* class**			***off-event* class**			**acc**
	prec	**rec**	**f1**	**prec**	**rec**	**f1**	
SVM (Modified)	**91.83**	**90.87**	**91.77**	**97.15**	**96.26**	**97.40**	**95.21**
SVM (All Feats. IG)	90.61	89.87	90.23	96.15	97.26	96.70	94.60
Trigger based	83.10	93.34	87.92	97.25	93.24	95.20	93.09

Table 5. % *on-event* overlap between the event classification systems.

$OnEvent_{overlap}$	**SVM (Modified)**	**LangModel (JM)**	**Trigger based**
SVM (Modified)	100%	-	-
LangModel (JM)	79.8%	100%	-
Trigger based	94.7%	81.3%	100%

Table 6. % *off-event* overlap between the event classification systems.

$OffEvent_{overlap}$	**SVM (Modified)**	**LangModel (JM)**	**Trigger based**
SVM (Modified)	100%	-	-
LangModel (JM)	13.86%	100%	-
Trigger based	36.62%	13.04%	100%

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 6), 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 treated this as a binary classification task and 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 are still encouraged by the fact that the uni-gram language modeling techniques achieved approximately 80% classification accuracy considering they require little or no feature engineering. From the results, we also observed that the performance of each approach for the *on-event* class varies considerably across the event types. This is a worrying aspect from an application point of view especially if the goal of it is to detect many different types of events. Additionally, we observed 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, sentence length and sentence position can also act as good indicators for this task. Finally our experiments also showed that the performance of the supervised approaches for this classification task are significantly affected by the type of dataset used for training. 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 discuss in this paper, namely, *event coreference 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 also wishes to thank the members of the Language Technology Research Group at the University of Melbourne and NICTA for their helpful discussions regarding this research.

Bibliography

[1] R. Saurí, R. Knippen, M. Verhagen, J. Pustejovsky, Evita: a robust event recognizer for qa systems, in: HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, 2005, pp. 700–707.

[2] W. Li, M. Wu, Q. Lu, W. Xu, C. Yuan, Extractive summarization using inter- and intra- event relevance, in: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Morristown, NJ, USA, 2006, pp. 369–376.

[3] L. Vanderwende, M. Banko, A. Menezes, Event-centric summary generation (2004).

[4] M. Wu, Investigations on event-based summarization, in: Proceedings of the COLING/ACL 2006 Student Research Workshop, 2006.

[5] E. Filatova, V. Hatzivassiloglou, Event-based extractive summarization, in: In Proceedings of ACL Workshop on Summarization, 2004, pp. 104 – 111.

[6] N. Daniel, D. Radev, T. Allison, Sub-event based multi-document summarization, in: Proceedings of the HLT-NAACL 03 on Text summarization workshop, Association for Computational Linguistics, Morristown, NJ, USA, 2003, pp. 9–16.

[7] G. Miller, Wordnet: a lexical database for english, Communications, ACM 38 (11) (1995) 39–41.

[8] C. Walker, S. Strassel, J. Medero, L. D. Consortium, ACE 2005 Multilingual Training Corpus, Linguistic Data Consortium, University of Pennsylvania, 2006.

[9] M. Naughton, N. Stokes, J. Carthy, Investigating statistical techniques for sentence-level event classification, in: Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), Coling 2008 Organizing Committee, Manchester, UK, 2008, pp. 617–624.

[10] R. Grishman, S. Huttunen, R. Yangarber, Real-time event extraction for infectious disease outbreaks, in: Proceedings of Human Language Technology Conference (HLT)., 2002, pp. 366–369.

[11] G. King, W. Lowe, 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) (2003) 617–642.

[12] M. Atkinson, J. Piskorski, B. Pouliquen, R. Steinberger, H. Tanev, V. Zavarella, Online-monitoring of security-related events, in: Coling 2008: Companion volume: Posters and Demonstrations, Coling 2008 Organizing Committee, Manchester, UK, 2008, pp. 1–4.

[13] Y. Yang, T. Pierce, J. Carbonell, 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, ACM Press New York, NY, USA, 1998, pp. 28–36.
[14] J. Allan, J. Carbonell, G. Doddington, J. Yamron, Y. Yang, Topic detection and tracking pilot study. final report (1998).
[15] Y. Yang, J. Carbonell, R. Brown, T. Pierce, B. Archibald, X. Liu, Learning approaches for detecting and tracking news events, IEEE Intelligent Systems 14 (4) (1999) 32–43.
[16] H. Murff, V. Patel, G. Hripcsak, D. Bates, Detecting adverse events for patient safety research: a review of current methodologies, Journal of Biomedical Informatics 36 (1/2) (2003) 131–143.
[17] G. Hripcsak, S. Bakken, P. Stetson, V. Patel, Mining complex clinical data for patient safety research: a framework for event discovery, Journal of Biomedical Informatics 36 (1/2) (2003) 120–130.
[18] D. Ahn, The stages of event extraction, in: Proceedings of the Workshop on Annotating and Reasoning about Time and Events, Association for Computational Linguistics, Sydney, Australia, 2006, pp. 1–8.
[19] J. Yamron, L. Gillick, P. van Mulbregt, S. Knecht, Statistical models of topical content, The Kluwer International Series on Information Retrieval (2002) 115–134.
[20] Y. Yang, T. Ault, T. Pierce, C. Lattimer, Improving text categorization methods for event tracking, in: Proceedings of the 23rd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM Press New York, NY, USA, 2000, pp. 65–72.
[21] F. Walls, H. Jin, S. Sista, R. Schwartz, Topic detection in broadcast news, in: In Proceedzngs of the DARPA Broadcast News Workshop, Morgan Kaufmann Publishers Inc., 1999, pp. 193 – 198.
[22] J. Schultz, M. Liberman, Topic detection and tracking using idf weighted cosine coefficient, in: In Proceedzngs of the DARPA Broadcast News Workshop, Morgan Kaufmann Publishers Inc., 1999, pp. 189–192.
[23] R. Schwartz, T. Imai, L. Nguyen, J. Makhoul, A maximum likelihood model for topic classification, in: Proceedings of Eurospeech, 1997, pp. 1455–1458.
[24] M. Sahami, S. Dumais, D. Heckerman, E. Horvitz, A bayesian approach to filtering junk E-mail, in: Learning for Text Categorization: Papers from the 1998 Workshop, AAAI Technical Report WS-98-05, Madison, Wisconsin, 1998.
[25] R. Segal, J. Crawford, J. Kephart, B. Leiba, Spamguru: An enterprise antispam filtering system (2004).
[26] R. Segal, J. Kephart, Incremental learning in swiftFile, in: Proc. 17th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA, 2000, pp. 863–870.
[27] M. Aery, S. Chakravarthy, emailsift: mining-based approaches to email classification, in: Proceedings of the 27th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM Press New York, NY, USA, 2004, pp. 580–581.
[28] M. Dredze, T. Lau, N. Kushmerick, Automatically classifying emails into activities, in: Proceedings of the 11th international conference on Intelligent user interfaces, ACM, New York, NY, USA, 2006, pp. 70–77.

[29] J. Cohen, A coeficient of agreement for nominal scales., Educational and Psychological Measurement 20 (1) (1960) 37–46.
[30] T. Joachims, 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, Springer Verlag, Heidelberg, DE, Chemnitz, DE, 1998, pp. 137–142.
[31] J. Platt, Fast training of support vector machines using sequential minimal optimization, Advances in kernel methods: support vector learning (1999) 185–208.
[32] I. Witten, E. Frank, Data mining: practical machine learning tools and techniques with Java implementations, Morgan Kaufmann Publishers Inc., 2000.
[33] M. Porter, An algorithm for suffix stripping, Readings in Information Retrieval (1997) 313–316.
[34] J. Curran, S. Clark, J. Bos, Linguistically motivated large-scale nlp with c & c and boxer, in: Proceedings of the ACL 2007 Demonstrations Session (ACL-07 demo), 2007, pp. 29–32.
[35] C. Zhai, J. Lafferty, A study of smoothing methods for language models applied to ad hoc information retrieval, in: Research and Development in Information Retrieval, 2001, pp. 334–342.
[36] W. Kraaij, M. Spitters, Language models for topic tracking, in: B. Croft, J. Lafferty (Eds.), Language Models for Information Retrieval, Kluwer Academic Publishers, 2003.
[37] C. Manning, H. Schütze, Foundations of Statistical Natural Language Processing, MIT Press, 1999.
[38] J. Allan, R. Gupta, V. Khandelwal, Temporal summaries of new topics, in: Proceedings of the 24th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM Press New York, NY, USA, 2001, pp. 10–18.
[39] K. Ng, P. Kantor, An investigation of the preconditions for effective data fusion in IR: A pilot study, in: In the Proceedings of the 61th Annual Meeting of the American Society for Information Science, 1998.