Identification of Event Mentions and their Semantic Class

Steven Bethard
Department of Computer Science
University of Colorado at Boulder
430 UCB, Boulder, CO 80309, USA
steven.bethard@colorado.edu

James H. Martin
Department of Computer Science
University of Colorado at Boulder
430 UCB, Boulder, CO 80309, USA
james.martin@colorado.edu

Abstract

Complex tasks like question answering need to be able to identify events in text and the relations among those events. We show that this event identification task and a related task, identifying the semantic class of these events, can both be formulated as classification problems in a word-chunking paradigm. We introduce a variety of linguistically motivated features for this task and then train a system that is able to identify events with a precision of 82% and a recall of 71%. We then show a variety of analyses of this model, and their implications for the event identification task.

1 Introduction

Research in question answering, machine translation and other fields has shown that being able to recognize the important entities in a text is often a critical component of these systems. Such entity information gives the machine access to a deeper level of semantics than words alone can provide, and thus offers advantages for these complex tasks. Of course, texts are composed of much more than just sets of entities, and architectures that rely solely on word and entity-based techniques are likely to have difficulty with tasks that depend more heavily on event and temporal relations. Consider a question answering system that receives the following questions:

- Is Anwar al-Sadat still the president of Egypt?
- How did the linking of the Argentinean peso to the US dollar in 1991 contribute to economic crisis of Argentina in 2003?

Processing such questions requires not only knowing what the important people, places and other entities are, but also what kind of events they are involved in, the roles they play in those events, and the relations among those events. Thus, we suggest that identifying such events in a text should play an important role in systems that attempt to address questions like these.

Of course, to identify events in texts, we must define what exactly it is we mean by “event”. In this work, we adopt a traditional linguistic definition of an event that divides words into two aspectual types: states and events. States describe situations that are static or unchanging for their duration, while events describe situations that involve some internal structure. For example, predicates like *know* and *love* would be states because if we *know* (or *love*) someone for a period of time, we *know* (or *love*) that person at each point during the period. Predicates like *run* or *deliver a sermon* would be events because they are built of smaller dissimilar components: *run* includes raising and lowering of legs and *deliver a sermon* includes the various tongue movements required to produce words.

To better explain how we approach the task of identifying such events, we first discuss some past work on related tasks. Then we briefly discuss the characteristics of the TimeBank, a corpus containing event-annotated data. Next we present our formulation of event identification as a classification task and introduce the linguistic features that serve as input to the algorithm. Finally, we show the results of STEP (our “System for Textual Event Parsing”) which applies these techniques to the TimeBank data.

2 Related Efforts

Such aspectual distinctions have been alive and well in the linguistic literature since at least the late 60s (Vendler, 1967). However, the use of the

Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), pages 146–154,
Sydney, July 2006. ©2006 Association for Computational Linguistics

term *event* in natural language processing work has often diverged quite considerably from this linguistic notion. In the Topic Detection and Tracking (TDT) task, events were sets of documents that described "some unique thing that happens at some point in time" (Allan et. al., 1998). In the Message Understanding Conference (MUC), events were groups of phrases that formed a template relating participants, times and places to each other (Marsh and Perzanowski, 1997). In the work of Filatova and Hatzivassiloglou (2003), events consisted of a verb and two named-entities occurring together frequently across several documents on a topic.

Several recent efforts have stayed close to the linguistic definition of events. One such example is the work of Siegel and McKeown (2000) which showed that machine learning models could be trained to identify some of the traditional linguistic aspectual distinctions. They manually annotated the verbs in a small set of texts as either state or event, and then used a variety of linguistically motivated features to train machine learning models that were able to make the event/state distinction with 93.9% accuracy.

Another closely related effort was the Evita system, developed by Saurí et. al. (2005). This work considered a corpus of events called TimeBank, whose annotation scheme was motivated largely by the linguistic definitions of events. Saurí et. al. showed that a linguistically motivated and mainly rule-based algorithm could perform well on this task.

Our work draws from both the Siegel and McKeown and Saurí et. al. works. We consider the same TimeBank corpus as Saurí et. al., but apply a statistical machine learning approach akin to that of Siegel and McKeown. We demonstrate that combining machine learning techniques with linguistically motivated features can produce models from the TimeBank data that are capable of making a variety of subtle aspectual distinctions.

3 Events in the TimeBank

TimeBank (Pustejovsky, et. al. 2003b) consists of just under 200 documents containing 70,000 words; it is drawn from news texts from a variety of different domains, including newswire and transcribed broadcast news. These documents are annotated using the TimeML annotation scheme (Pustejovsky, et. al. 2003a), which aims to identify not just times and dates, but events and the temporal relations between these events.

Of interest here are the EVENT annotations, of which TimeBank 1.1 has annotated 8312. TimeBank annotates a word or phrase as an EVENT if it describes a situation that can "happen" or "occur", or if it describes a "state" or "circumstance" that "participate[s] in an opposition structure in a given text" (Pustejovsky, et. al. 2003b). Note that the TimeBank events are not restricted to verbs; nouns and adjectives denote events as well.

The TimeBank definition of event differs in a few ways from the traditional linguistic definition of event. TimeBank EVENTs include not only the normal linguistic events, but also some linguistic states, depending on the contexts in which they occur. For example[1], in the sentence *None of the people on board the airbus survived the crash* the phrase *on board* would be considered to describe an EVENT because that state changes in the time span covered by the text. Not all linguistic states become TimeBank EVENTs in this manner, however. For example, the state described by *New York is on the east coast* holds true for a time span much longer than the typical newswire document and would therefore not be labeled as an EVENT.

In addition to identifying which words in the TimeBank are EVENTs, the TimeBank also provides a semantic class label for each EVENT. The possible labels include OCCURRENCE, PERCEPTION, REPORTING, ASPECTUAL, STATE, I_STATE, I_ACTION, and MODAL, and are described in more detail in (Pustejovsky, et. al. 2003a).

We consider two tasks on this data:

(1) Identifying which words and phrases are EVENTs, and

(2) Identifying their semantic classes.

The next section describes how we turn these tasks into machine learning problems.

4 Event Identification as Classification

We view event identification as a classification task using a word-chunking paradigm similar to that used by Carreras et. al. (2002). For each word in a document, we assign a label indicating whether the word is inside or outside of an event. We use the standard B-I-O formulation of the word-chunking task that augments each class label with an indicator of whether the given word

[1] These examples are derived from (Pustejovsky, et. al. 2003b)

Word	*Event Label*	*Event Semantic Class Label*
The	O	O
company	O	O
’s	O	O
sales	O	O
force	O	O
applauded	B	B_I_ACTION
the	O	O
shake	B	B_OCCURRENCE
up	I	I_OCCURRENCE
.	O	O

Table 1: Event chunks for sentence (1)

is (B)eginning, (I)nside or (O)utside of a chunk (Ramshaw & Marcus, 1995). So, for example, under this scheme, sentence (1) would have its words labeled as in Table 1.

(1) The company’s sales force [EVENT(I_ACTION) applauded] the [EVENT(OCCURRENCE) shake up]

The two columns of labels in Table 1 show how the class labels differ depending on our task. If we’re interested only in the simple event identification task, it’s sufficient to know that *applauded* and *shake* both begin events (and so have the label B), *up* is inside an event (and so has the label I), and all other words are outside events (and so have the label O). These labels are shown in the column labeled Event Label. If in addition to identifying events, we also want to identify their semantic classes, then we need to know that *applauded* begins an intentional action event (B_I_ACTION), *shake* begins an occurrence event (B_OCCURRENCE), *up* is inside an occurrence event (I_OCCURRENCE), and all other words are outside of events (O). These labels are shown in the column labeled Event Semantic Class Label. Note that while the eight semantic class labels in the TimeBank could potentially introduce as many as 8 • 2 + 1 = 17 chunk labels, not all types of events appear as multi-word phrases, so we see only 13 of these labels in our data.

5 Classifier Features

Having cast the problem as a chunking task, our next step is to select and represent a useful set of features. In our case, since each classification instance is a word, our features need to provide the information that we deem important for recognizing whether a word is part of an event or not. We consider a number of such features, grouped into feature classes for the purposes of discussion.

5.1 Text feature

This feature is just the textual string for the word.

5.2 Affix features

These features attempt to isolate the potentially important subsequences of characters in the word. These are intended to identify affixes that have a preference for different types of events.

Affixes: These features identify the first three and four characters of the word, and the last three and four characters of the word.

Nominalization suffix: This feature indicates which of the suffixes typically associated with nominalizations – *ing(s), ion(s), ment(s),* and *nce(s)* – the word ends with. This overlaps with the Suffixes feature, but allows the classifier to more easily treat nominalizations specially.

5.3 Morphological features

These features identify the various morphological variants of a word, so that, for example, the words *resist*, *resisted* and *resistance* can all be identified as the same basic event type.

Morphological stem: This feature gives the base form of the word, so for example, the stem of *assisted* is *assist* and the stem of *investigations* is *investigation*. Stems are identified with a lookup table from the University of Pennsylvania of around 300,000 words.

Root verb: This feature gives the verb from which the word is derived. For example, *assistance* is derived from *assist* and *investigation* is derived from *investigate*. Root verbs are identified with an in-house lookup table of around 5000 nominalizations.

5.4 Word class features

These features attempt to group the words into different types of classes. The intention here is to identify correlations between classes of words and classes of events, e.g. that events are more likely to be expressed as verbs or in verb phrases than they are as nouns.

Part-of-speech: This feature contains the word’s part-of-speech based on the Penn Treebank tag set. Part-of-speech tags are assigned by the MX-POST maximum-entropy based part-of-speech tagger (Ratnaparkhi, 1996).

Syntactic-chunk label: The value of this feature is a B-I-O style label indicating what kind of syntactic chunk the word is contained in, e.g. noun phrase, verb phrase, or prepositional phrase. These are assigned using a word-chunking SVM-based system trained on the CoNLL-2000 data[2] (which uses the lowest nodes of the Penn TreeBank syntactic trees to break sentences into base phrases).

Word cluster: This feature indicates which verb or noun cluster the word is a member of. The clusters were derived from the co-occurrence statistics of verbs and their direct objects, in the same manner as Pradhan et. al. (2004). This produced 128 clusters (half verbs, half nouns) covering around 100,000 words.

5.5 Governing features

These features attempt to include some simple dependency information from the surrounding words, using the dependency parses produced by Minipar[3]. These features aim to identify events that are expressed as phrases or that require knowledge of the surrounding phrase to be identified.

Governing light verb: This feature indicates which, if any, of the light verbs *be*, *have*, *get*, *give*, *make*, *put*, and *take* governs the word. This is intended to capture adjectival predicates such as *may be ready*, and nominal predicates such as *make an offer*, where *ready* and *offer* should be identified as events.

Determiner type: This feature indicates the type of determiner a noun phrase has. If the noun phrase has an explicit determiner, e.g. *a*, *the* or *some*, the value of this feature is the determiner itself. We use the determiners themselves as feature values here because they form a small, closed class of words. For open-class determiner-like modifiers, we instead group them into classes. For noun phrases that are explicitly quantified, like *a million dollars*, the value is CARDINAL, while for noun phrases modified by other possessive noun phrases, like *Bush's real objectives*, the value is GENITIVE. For noun phrases without a determiner-like modifier, the value is PROPER_NOUN, BARE_PLURAL or BARE_SINGULAR, depending on the noun type.

Subject determiner type: This feature indicates for a verb the determiner type (as above) of its subject. This is intended to distinguish generic sentences like *Cats have fur* from non-generics like *The cat has fur*.

5.6 Temporal features

These features try to identify temporal relations between words. Since the duration of a situation is at the core of the TimeBank definition of events, features that can get at such information are particularly relevant.

Time chunk label: The value of this feature is a B-I-O label indicating whether or not this word is contained in a temporal annotation. The temporal annotations are produced by a word-chunking SVM-based system trained on the temporal expressions (TIMEX2 annotations) in the TERN 2004 data[4]. In addition to identifying expressions like *Monday* and *this year*, the TERN data identifies event-containing expressions like *the time she arrived at her doctor's office*.

Governing temporal: This feature indicates which kind of temporal preposition governs the word. Since the TimeBank is particularly interested in which events start or end within the time span of the document, we consider prepositions likely to indicate such a change of state, including *after*, *before*, *during*, *following*, *since*, *till*, *until* and *while*.

Modifying temporal: This feature indicates which kind of temporal expression modifies the word. Temporal expressions are recognized as above, and the type of modification is either the preposition that joins the temporal annotation to the word, or ADVERBIAL for any non-preposition modification. This is intended to capture that modifying temporal expressions often indicate event times, e.g. *He ran the race in an hour*.

5.7 Negation feature

This feature indicates which negative particle, e.g. *not*, *never*, etc., modifies the word. The idea is based Siegel and McKeown’s (2000) findings which suggested that in some corpora states occur more freely with negation than events do.

5.8 WordNet hypernym features

These features indicate to which of the WordNet noun and verb sub-hierarchies the word belongs.

[2] http://cnts.uia.ac.be/conll2000/

[3] http://www.cs.ualberta.ca/~lindek/minipar.htm

[4] http://timex2.mitre.org/tern.html

Rather than include all of the thousands of different sub-hierarchies in WordNet, we first selected the most useful candidates by looking at the overlap with WordNet and our training data. For each hierarchy in WordNet, we considered a classifier that labeled all words in that hierarchy as events, and all words outside of that hierarchy as non-events[5]. We then evaluated these classifiers on our training data, and selected the ten with the highest F-measures. This resulted in selecting the following synsets:

- noun: state
- noun: psychological feature
- noun: event
- verb: think, cogitate, cerebrate
- verb: move, displace
- noun: group, grouping
- verb: act, move
- noun: act, human action, human activity
- noun: abstraction
- noun: entity

The values of the features were then whether or not the word fell into the hierarchy defined by each one of these roots. Note that since there are no WordNet senses labeled in our data, we accept a word as falling into one of the above hierarchies if any of its senses fall into that hierarchy.

6 Classifier Parameters

The features described in the previous section give us a way to provide the learning algorithm with the necessary information to make a classification decision. The next step is to convert our training data into sets of features, and feed these classification instances to the learning algorithm. For the learning task, we use the TinySVM[6] support vector machine (SVM) implementation in conjunction with YamCha[7] (Kudo & Matsumoto, 2001), a suite for general-purpose chunking.

YamCha has a number of parameters that define how it learns. The first of these is the window width of the "sliding window" that it uses. A sliding window is a way of including some of the context when the classification decision is made for a word. This is done by including the features of preceding and following words in addition to the features of the word to be classified. To illustrate this, we consider our earlier example, now augmented with some additional features in Table 2.

Word	*POS*	*Stem*	*Label*
The	DT	the	O
company	NN	company	O
's	POS	's	O
sales	NNS	sale	O
force	NN	force	O
applauded	VBD	applaud	B
The	DT	the	O
shake	NN	shake	B
up	RP	up	
.	.	.	

Table 2: A window of word features

To classify *up* in this scenario, we now look not only at its features, but at the features of some of the neighboring words. For example, if our window width was 1, the feature values we would use for classification would be those in the outlined box, that is, the features of *shake*, *up* and the sentence final period. Note that we do not include the classification labels for either *up* or the period since neither of these classifications is available at the time we try to classify *up*. Using such a sliding window allows YamCha to include important information, like that *up* is preceded by *shake* and that *shake* was identified as beginning an event.

In addition to the window width parameter, YamCha also requires values for the following three parameters: the penalty for misclassification (C), the kernel's polynomial degree, and the method for applying binary classifiers to our multi-class problem, either pair-wise or one-vs-rest. In our experiments, we chose a one-vs-rest multi-class scheme to keep training time down, and then tried different variations of all the other parameters to explore a variety of models.

7 Baseline Models

To be able to meaningfully evaluate the models we train, we needed to establish a reasonable baseline. Because the majority class baseline would simply label every word as a non-event, we introduce two baseline models that should be more reasonable: Memorize and Sim-Evita.

[5] We also considered the reverse classifiers, which classified all words in the hierarchy as non-events and all words outside the hierarchy as events.

[6] http://chasen.org/~taku/software/TinySVM/

[7] http://chasen.org/~taku/software/yamcha/

The Memorize baseline is essentially a lookup table – it memorizes the training data. This system assigns to each word the label with which it occurred most frequently in the training data, or the label O (not an event) if the word never occurred in the training data.

The Sim-Evita model is our attempt to simulate the Evita system (Saurí et. al. 2005). As part of its algorithm, Evita includes a check that determines whether or not a word occurs as an event in TimeBank. It performs this check even when evaluated on TimeBank, and thus though Evita reports 74% precision and 87% recall, these numbers are artificially inflated because the system was trained and tested on the same corpus. Thus we cannot directly compare our results to theirs. Instead, we simulate Evita by taking the information that it encodes as rules, and encoding this instead as features which we provide to a YamCha-based system.

Saurí et. al. (2005) provides a description of Evita's rules, which, according to the text, are based on information from lexical stems, part of speech tags, syntactic chunks, weak stative predicates, copular verbs, complements of copular predicates, verbs with bare plural subjects and WordNet ancestors. We decided that the following features most fully covered the same information:

- Text
- Morphological stem
- Part-of-speech
- Syntactic-chunk label
- Governing light verb
- Subject determiner type
- WordNet hypernyms

We also decided that since Evita does not consider a word-window around the word to be classified, we should set our window size parameter to zero.

Because our approximation of Evita uses a feature-based statistical machine learning algorithm instead of the rule-based Evita algorithm, it cannot predict how well Evita would perform if it had not used the same data for training and testing. However, it can give us an approximation of how well a model can perform using information similar to that of Evita.

8 Results

Having decided on our feature space, our learning model, and the baselines to which we will compare, we now describe the results of our models on the TimeBank. We selected a stratified sample of 90% of the TimeBank data for a training set, and reserved the remaining 10% for testing[8].

We consider three evaluation measures: precision, recall and F-measure. Precision is defined as the number of B and I labels our system identifies correctly, divided by the total number of B and I labels our system predicted. Recall is defined as the number of B and I labels our system identifies correctly, divided by the total number of B and I labels in the TimeBank data. F-measure is defined as the geometric mean of precision and recall[9].

To determine the best parameter settings for the models, we performed cross-validations on our training data, leaving the testing data untouched. We divided the training data randomly into five equally-sized sections. Then, for each set of parameters to be evaluated, we determined a cross-validation F-measure by averaging the F-measures of five runs, each tested on one of the training data sections and trained on the remaining training data sections. We selected the parameters of the model that had the best cross-validation F-measure on the training data as the parameters for the rest of our experiments. For the simple event identification model this selected a window width of 2, polynomial degree of 3 and C value of 0.1, and for the event and class identification model this selected a window width of 1, polynomial degree of 1 and C value 0.1. For the Sim-Evita simple event identification model this selected a degree of 2 and C value of 0.01, and for the Sim-Evita event and class identification model, this selected a degree of 1 and C value of 1.0.

Having selected the appropriate parameters for our learning algorithm, we then trained our SVM models on the training data. Table 3 presents the results of these models on the test data. Our model (named STEP above for "System for Tex-

[8] The testing documents were: APW19980219.0476, APW19980418.0210, NYT19980206.0466, PRI19980303.2000.2550, ea980120.1830.0071, and the wsj_XXXX_orig documents numbered 0122, 0157, 0172, 0313, 0348, 0541, 0584, 0667, 0736, 0791, 0907, 0991 and 1033.

[9] $F = \frac{2 \cdot P \cdot R}{P + R}$

	Event Identification			Event and Class Identification		
Model	*Precision*	*Recall*	*F*	*Precision*	*Recall*	*F*
Memorize	0.806	0.557	0.658	0.640	0.413	0.502
Sim-Evita	0.812	0.659	0.727	0.571	0.459	0.509
STEP	0.820	0.706	0.759	0.667	0.512	0.579

Table 3: Overall results for both tasks

	Event Identification				Event and Class Identification			
	%	*Precision*	*Recall*	*F*	*%*	*Precision*	*Recall*	*F*
Verbs	59	0.864	0.903	0.883	59	0.714	0.701	0.707
Nouns	28	0.729	0.432	0.543	28	0.473	0.261	0.337

Table 4: Results by word class for both tasks

	%	*Precision*	*Recall*	*F*
B	92	0.827	0.737	0.779
I	8	0.679	0.339	0.452
B Occurrence	44	0.633	0.727	0.677
B State	14	0.519	0.136	0.215
B Reporting	11	0.909	0.779	0.839
B Istate	10	0.737	0.378	0.500
B Iaction	10	0.480	0.174	0.255
I State	7	0.818	0.173	0.286
B Aspectual	3	0.684	0.684	0.684

Table 5: Results by label

tual Event Parsing") outperforms both baselines on both tasks. For simple event identification, the main win over both baselines is an increased recall. Our model achieves a recall of 70.6%, about 5% better than our simulation of Evita, and nearly 15% better than the Memorize baseline. For event and class identification, the win is again in recall, though to a lesser degree. Our system achieves a recall of 51.2%, about 5% better than Sim-Evita, and 10% better than Memorize. On this task, we also achieve a precision of 66.7%, about 10% better than the precision of Sim-Evita. This indicates that the model trained with no context window and using the Evita-like feature set was at a distinct disadvantage over the model which had access to all of the features.

Table 4 and Table 5 show the results of our systems on various sub-tasks, with the "%" column indicating what percent of the events in the test data each subtask contained. Table 4 shows that in both tasks, we do dramatically better on verbs than on nouns, especially as far as recall is concerned. This is relatively unsurprising – not only is there more data for verbs (59% of event words are verbs, while only 28% are nouns), but our models generally do better on words they have seen before, and there are many more nouns we have not seen than there are verbs.

Table 5 shows how well we did individually on each type of label. For simple event identification (the top two rows) we can see that we do substantially better on B labels than on I labels, as we would expect since 92% of event words are labeled B. The label-wise performance for the event and class identification (the bottom seven rows) is more interesting. Our best performance is actually on Reporting event words, even though the data is mainly Occurrence event words. One reason for this is that instances of the word *said* make up about 60% of Reporting event words in the TimeBank. The word *said* is relatively easy to get right because it comes with by far the most training data[10], and because it is almost always an event: 98% of the time in the TimeBank, and 100% of the time in our test data.

To determine how much each of the feature sets contributed to our models we also performed a pair of ablation studies. In each ablation study, we trained a series of models on successively fewer feature sets, removing the least important feature set each time. The least important feature set was determined by finding out which feature set's removal caused the smallest drop in F-measure. The result of this process was a list of our feature sets, ordered by importance. These lists are given for both tasks in Table 6, along with the precision, recall and F-measures of the various corresponding models. Each row in Table 6 corresponds to a model trained on the feature sets named in that row and all the rows below it. Thus, on the top row, no feature sets have been removed, and on the bottom row only one feature set remains.

[10] The word "said" has over 600 instances in TimeBank. The word with the next most instances has just over 200

Event Identification				Event and Class Identification			
Feature set	*Precision*	*Recall*	*F*	*Feature set*	*Precision*	*Recall*	*F*
Governing	0.820	0.706	0.759	Governing	0.667	0.512	0.579
Negation	0.824	0.713	0.765	Temporal	0.675	0.513	0.583
Affix	0.826	0.715	0.766	Negation	0.672	0.510	0.580
WordNet	0.818	0.723	0.768	Morphological	0.670	0.509	0.579
Temporal	0.820	0.729	0.772	Text	0.671	0.505	0.576
Morphological	0.816	0.727	0.769	WordNet	0.679	0.497	0.574
Text	0.816	0.697	0.752	Word class	0.682	0.474	0.559
Word class	0.719	0.677	0.697	Affix	0.720	0.421	0.531

Table 6: Ablations for both tasks. For each task, the least important feature sets appear at the top of the table, and most important feature sets appear at the bottom. For each row, the precision, recall and F-measure indicate the scores of a model trained with only the feature sets named in that row and the rows below it.

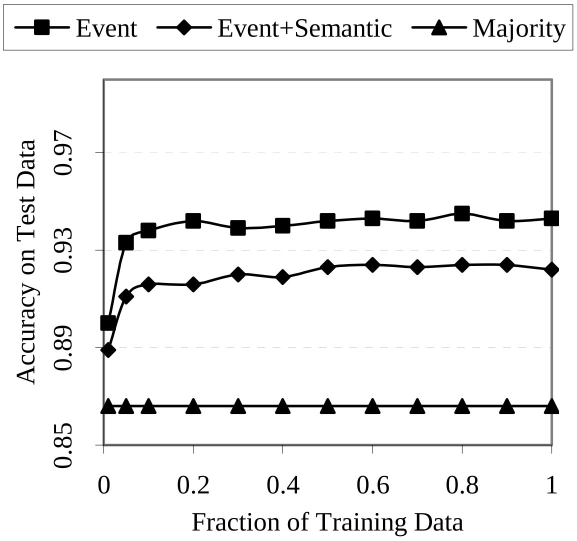

Figure 1: Learning Curves

So, for example, in the simple event identification task, we see that the Governing, Negation, Affix and WordNet features are hurting the classifier somewhat – a model trained without these features performs at an F-measure of 0.772, more than 1% better than a model including these features. In contrast, we can see that for the event and semantic class identification task, the WordNet and Affix features are actually among the most important, with only the Word class features accompanying them in the top three. These ablation results suggest that word class, textual, morphological and temporal information is most useful for simple event identification, and affix, WordNet and negation information is only really needed when the semantic class of an event must also be identified.

The last thing we investigated was the effect of additional training data. To do so, we trained the model on increasing fractions of the training data, and measured the classification accuracy on the testing data of each of the models thus trained. The resulting graph is shown in Figure 1. The Majority line indicates the classifier accuracy when the classifier always guesses majority class, that is, (O)utside of an event. We can see from the two learning curves that even with only the small amount of data available in the TimeBank, our models are already reaching the level part of the learning curve at somewhere around 20% of the data. This suggests that, though additional data may help somewhat in the data sparseness problem, substantial further progress on this task will require new, more descriptive features.

9 Conclusions

In this paper, we showed that statistical machine learning techniques can be successfully applied to the problem of identifying fine-grained events in a text. We formulated this task as a statistical classification task using a word-chunking paradigm, where words are labeled as beginning, inside or outside of an event. We introduced a variety of relevant linguistically-motivated features, and showed that models trained in this way could perform quite well on the task, with a precision of 82% and a recall of 71%. This method extended to the task of identifying the semantic class of an event with a precision of 67% and a recall of 51%. Our analysis of these models indicates that while the simple event identification task can be approached with mostly simple text and word-class based features, identifying the semantic class of an event requires features that encode more of the semantic context of the words. Finally, our training curves suggest that future research in this area should focus primarily on identifying more discriminative features.

10 Acknowledgments

This work was partially supported by a DHS fellowship to the first author and by ARDA under AQUAINT project MDA908-02-C-0008. Computer time was provided by NSF ARI Grant #CDA-9601817, NSF MRI Grant #CNS-0420873, NASA AIST grant #NAG2-1646, DOE SciDAC grant #DE-FG02-04ER63870, NSF sponsorship of the National Center for Atmospheric Research, and a grant from the IBM Shared University Research (SUR) program. Any opinions, findings, or recommendations are those of the authors and do not necessarily reflect the views of the sponsors. Particular thanks go to Wayne Ward and Martha Palmer for many helpful discussions and comments.

References

James Allan, Jaime Carbonell, George Dodding-ton, Jonathan Yamron and Yiming Yang. 1998. Topic Detection and Tracking Pilot Study: Final Report. In: Proceedings of DARPA Broadcast News Transcription and Understanding Workshop.

Xavier Carreras, Lluís Màrquez and Lluís Padró. Named Entity Extraction using AdaBoost. 2002. In Proceedings of CoNNL-2002.

Elena Filatova and Vasileios Hatzivassiloglou. Domain-Independent Detection, Extraction, and Labeling of Atomic Events. 2003. In the Proceedings of Recent Advances in Natural Language Processing Conference, September 2003.

Taku Kudo and Yuji Matsumoto. 2001. Chunking with support vector machines. In Proceedings of NAACL 2001.

Elaine Marsh and Dennis Perzanowski. 1997. MUC-7 evaluation of IE technology: Over-view of results. In Proceedings of the Seventh MUC.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H. Martin and Daniel Jurafsky. 2004. Shallow Semantic Parsing using Support Vector Machines. In Proceedings of HLT/NAACL 2004.

James Pustejovsky, José Castaño, Robert Ingria, Roser Saurí, Robert Gaizauskas, Andrea Setzer and Graham Katz. TimeML: 2003a. Robust Specification of Event and Temporal Expressions in Text. In Proceedings of the Fifth International Workshop on Computational Semantics (IWCS-5)

James Pustejovsky, Patrick Hanks, Roser Saurí, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro and Marcia Lazo. 2003b. The TIMEBANK Corpus. In Proceedings of Corpus Linguistics 2003, 647-656.

Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text Chunking using Transformation-Based Learning. In Proceedings of the ACL Third Workshop on Very Large Corpora. 82-94.

Adwait Ratnaparkhi. 1996. A maximum entropy part-of-speech tagger. In Proceedings of EMNLP 1996.

Roser Saurí, Robert Knippen, Marc Verhagen and James Pustejovsky 2005. Evita: A Robust Event Recognizer For QA Systems. In Proceedings of HLT-EMNLP 2005.

Eric V. Siegel and Kathleen R. McKeown. Learning Methods to Combine Linguistic Indicators: Improving Aspectual Classification and Revealing Linguistic Insights. *Computational Linguistics*, 26(4):595 627.

Zeno Vendler. 1967. Verbs and times. In *Linguistics and Philosophy*. Cornell University Press, Ithaca, New