Class	Algor.	Prec.	Recall	F-Score
Short	**SVM**	0.707	0.606	**0.653**
	NB	0.567	0.768	0.652
	C4.5	0.571	0.600	0.585
Long	**SVM**	0.793	0.857	**0.823**
	NB	0.834	0.665	0.740
	C4.5	0.765	0.743	0.754

Table 5: Test Performance of Three Algorithms.

4.2 Data

The original annotated data can be straightforwardly transformed for this binary classification task. For each event annotation, the most likely (mean) duration is calculated first by averaging (the logs of) its lower and upper bound durations. If its most likely (mean) duration is less than a day (about 11.4 in the natural logarithmic scale), it is assigned to the "short" event class, otherwise it is assigned to the "long" event class. (Note that these labels are strictly a convenience and not an analysis of the meanings of "short" and "long".)

We divide the total annotated non-WSJ data (2132 event instances) into two data sets: a training data set with 1705 event instances (about 80% of the total non-WSJ data) and a held-out test data set with 427 event instances (about 20% of the total non-WSJ data). The WSJ data (156 event instances) is kept for further test purposes (see Section 4.4).

4.3 Experimental Results (non-WSJ)

Learning Algorithms. Three supervised learning algorithms were evaluated for our binary classification task, namely, Support Vector Machines (SVM) (Vapnik, 1995), Naïve Bayes (NB) (Duda and Hart, 1973), and Decision Trees C4.5 (Quinlan, 1993). The Weka (Witten and Frank, 2005) machine learning package was used for the implementation of these learning algorithms. Linear kernel is used for SVM in our experiments.

Each event instance has a total of 18 feature values, as described in Section 3, for the event only condition, and 30 feature values for the local context condition when $n = 2$. For SVM and C4.5, all features are converted into binary features (6665 and 12502 features).

Results. 10-fold cross validation was used to train the learning models, which were then tested on the unseen held-out test set, and the performance (including the precision, recall, and F-score[1] for each class) of the three learning algorithms is shown in Table 5. The significant measure is overall precision, and this is shown for the three algorithms in Table 6, together with human agreement (the upper bound of the learning task) and the baseline.

Algorithm	Precision
Baseline	59.0%
C4.5	69.1%
NB	70.3%
SVM	**76.6%**
Human Agreement	87.7%

Table 6: Overall Test Precision on non-WSJ Data.

We can see that among all three learning algorithms, SVM achieves the best F-score for each class and also the best overall precision (76.6%). Compared with the baseline (59.0%) and human agreement (87.7%), this level of performance is very encouraging, especially as the learning is from such limited training data.

Feature Evaluation. The best performing learning algorithm, SVM, was then used to examine the utility of combinations of four different feature sets (i.e., event, local context, syntactic, and WordNet hypernym features). The detailed comparison is shown in Table 7.

We can see that most of the performance comes from event word or phrase itself. A significant improvement above that is due to the addition of information about the subject and object. Local context does not help and in fact may hurt, and hypernym information also does not seem to help[2]. It is of interest that the most important information is that from the predicate and arguments describing the event, as our linguistic intuitions would lead us to expect.

4.4 Test on WSJ Data

Section 4.3 shows the experimental results with the learned model trained and tested on the data with the same genre, i.e., non-WSJ articles.

In order to evaluate whether the learned model can perform well on data from different news genres, we tested it on the unseen WSJ data (156 event instances). The performance (including the precision, recall, and F-score for each class) is shown in Table 8. The precision (75.0%) is very close to the test performance on the non-WSJ

[1] F-score is computed as the harmonic mean of the precision and recall: F = (2*Prec*Rec)/(Prec+Rec).

[2] In the "Syn+Hyper" cases, the learning algorithm with and without local context gives identical results, probably because the other features dominate.