Automatic Time Expression Labeling for English and Chinese Text

Kadri Hacioglu, Ying Chen, and Benjamin Douglas

Center for Spoken Language Research,
University of Colorado at Boulder,
Boulder, Colorado 80309

Abstract. In this paper, we describe systems for automatic labeling of time expressions occurring in English and Chinese text as specified in the ACE Temporal Expression Recognition and Normalization (TERN) task. We cast the chunking of text into time expressions as a tagging problem using a bracketed representation at token level, which takes into account embedded constructs. We adopted a left-to-right, token-by-token, discriminative, deterministic classification scheme to determine the tags for each token. A number of features are created from a predefined context centered at each token and augmented with decisions from a rule-based time expression tagger and/or a statistical time expression tagger trained on different type of text data, assuming they provide complementary information. We trained one-versus-all multi-class classifiers using support vector machines. We participated in the TERN 2004 recognition task and achieved competitive results.

1 Introduction

Extraction of temporal expressions from an input text is considered a very important step in several natural language processing tasks; namely, information extraction, question answering (QA), summarization etc. ([Mani 2004]). For example, in the summarization task, temporal expressions can be used to establish a time line for all events mentioned in multiple documents for a coherent summarization. Recently, there has been growing interest in addressing temporal questions in QA systems ([Schilder and Habel 2003]; [Saquete et. al. 2004]). In those systems, a highly accurate temporal expression recognizer or tagger (statistical or rule-based) is required for effective treatment of temporal questions yielding high-quality end-to-end system performance.

An official evaluation, sponsored by the DARPA automatic content extraction (ACE) program, has been organized by MITRE and NIST for time expression recognition and normalization (TERN) in 2004. The TERN task requires the recognition of a broad range of temporal expressions in the text and normalization of those expressions according to ([Ferro et. al. 2004]). The annotation is intended to mark information in the source text that mentions *when* something happened, *how long* something lasted, or *how often* something occurs. Temporal

A. Gelbukh (Ed.): CICLing 2005, LNCS 3406, pp. 548–559, 2005.

Table 1. Word and document statistics of TERN corpus. Numbers in parentheses indicate the total number of documents

	Development	Test
English	265K (767)	55K (192)
Chinese	147K (466)	67K (256)

expressions in text vary from explicit references, e.g. *June 1, 1995*, to implicit references, e.g. *last summer*, to durations, e.g. *four years*, to sets, e.g. *every month*, and to event-anchored expressions, e.g. *a year after the earthquake*.

We participated in the recognition task for both English and Chinese text. Here, we adopt an end-to-end statistical approach by identifying three sub-tasks; (i) data representation, (ii) feature engineering and (iii) model learning. The original XML representation of time expressions is changed into a bracketed representation by assigning tags to each token. The bracketed representation is chosen to account for embedded structures in the time expressions. Tags indicate whether a token is inside a time expression, it begins a time expression, it ends a time expression, or it is outside a time expression. Several lexical, syntactic and semantic features are chosen based on intuition, experience and data analysis. One-versus-all classifiers ([Allwein et. al 2000]) are trained using support vector machines (SVMs) ([Vapnik 1995]; [Burges 1998]) and all system settings (e.g. polynomial degree, regularization parameter and context window) are optimized using a held-out data set. Our labeling (or tagging) scheme being language independent yields almost identical systems for both English and Chinese languages. We test the final systems on the official evaluation data set and report competitive results.

The paper is organized as follows. In the next section, we describe the TERN corpus. The English system is described in Section 3. Section 4 introduces the Chinese system. Experimental set-up and results are presented in Section 5. Section 6 concludes the paper with some remarks and possible future work.

2 Description of Data

The TERN corpus is composed of text selected from broadcast news programs, newspapers and newswire reports. It is available in two different sets for both Chinese and English languages; one set is for system training/development and the other set is for evaluation. Table 1 summarizes word and document statistics of the TERN corpus.

3 English System

In this section we describe the TERN system developed for English text. We cast the time expression extraction as a supervised tagging problem. We illustrate this for the sentence *That's 30 percent more than the same period a year ago.*,

which contains an embedded time expression. This text appears in the training set and it is tagged with the xml-style TIMEX2 tags in the following way:

> That's 30 percent more than <TIMEX2> the same period <TIMEX2> a year ago </TIMEX2> </TIMEX2>.

The sentence is converted to a vertical token-level representation by using bracketed representation to incorporate the embedded structure as illustrated below:

That	O
's	O
30	O
percent	O
more	O
than	O
the	(*
same	*
period	*
a	(*
year	*
ago	*))
.	O

The time expressions in the example sentence are enclosed between the brackets. Each word is assigned a tag depending on its position with respect to the time expressions in the sentence. In the example, "O" indicates an outside word (or token), "(*" indicates the beginning of a time expression, "*" indicates a word inside a time expression and "*))" indicates a word that ends the embedded time expression. This representation requires a sentence segmenter and a tokenizer for a given raw document that contains several sentences. Therefore, the TERN data is first segmented into sentences and next tokenized using MXTERMINATOR[1] ([Reynar and Ratnaparkhi 1997]) and a slightly modified version of the Penn Tree Bank tokenizer [2], respectively. Finally, the original TIMEX2 tags are converted into the bracketed representations illustrated earlier. In doing so, the TERN training data is organized as one-token per line with sentences separated by blank lines. This data is passed to other modules for the creation of token specific features.

We define a number of features for each token. Our features can be grouped into four broad classes; lexical, syntactic, semantic and external features. The lexical features are the token itself, its lower-case version, its part of speech tag, a set of features that indicates a specific token pattern (e.g. is hyphenated or not, is a number etc.) and its frequency (e.g. Rare/Frequent/Unknown) with respect to a lexicon (with counts) created from the training data. The syntactic

[1] `http://www.cis.upenn.edu/~adwait/statnlp.html`
[2] `www.cis.upenn.edu/~treebank/tokenization.html`

features that we extract are base phrase chunks ([Ramhsaw and Marcus 1995]; [Kudo and Matsumato 2000]) represented using IOB2 tags as considered in the paper ([Sang and Veenstra 1999]). The head words and dependency relations between the tokens and their respective heads are considered as semantic features. We use a part-of-speech (POS) tagger, trained in-house, to determine the POS tag for each word. This tagger is based on the Yamcha SVM toolkit [3] and trained on a relatively large portion of the Penn TreeBank. Similarly, the base phrase chunks are obtained using an in-house trained SVM-based chunker. The dependency features are assembled from the output of Minipar[4] ([Lin 1998]), a rule-based dependency parser. In addition to those features, we use external features as the decisions from a rule-based time expression tagger (distributed by TERN organizers [5] which covers many of the types of time expressions contained in the TIMEX2 2001 guidelines) and BBN IdentiFinder ([Bikel et. al 1999]). Summarizing, the following features are used within a predefined, finite-size sliding window:

- tokens
- lower-cased tokens
- POS tags
- token pattern flags
- token frequency
- base phrase chunks
- head words
- dependency relations
- rule-based time expression tags
- BBN-Identifinder time expression tags
- previous time expression decisions

A total of 10 one-versus-all SVM classifiers were trained using a polynomial kernel of degree 2. The regularization parameter of SVMs was set to C=1.0. The class labels are illustrated below:

((*, ((*), (*, (*), (*)), *, *), *)), *))), O

The general architecture of the English system is shown in Figure 1. After the SVM classification we employ a simple post-processing algorithm to maintain the consistency of bracketing that might have been violated due to tagging errors. In the following, we summarize the steps taken in the system for the extraction of time expressions:

Step 1. Sentence segmentation
Step 2. Tokenization
Step 3. Pattern and frequency checking

[3] http://cl.aist-nara.ac.jp/taku-ku/software/TinySVM
http://cl.aist-nara.ac.jp/taku-ku/software/yamcha
[4] http://www.cs.ualberta.ca/~lindek/downloads.htm
[5] http://timex2.mitre.org/taggers/timex2_taggers.html

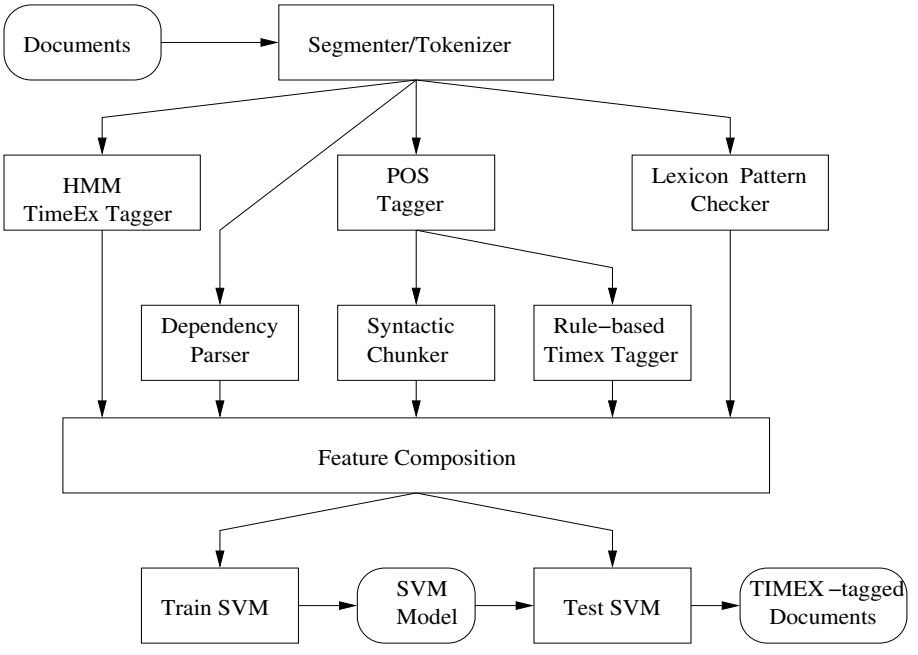

Fig. 1. General system architecture for English automatic time expression labeler (see text for details)

Step 4. Dependency Parsing
Step 5. Third-party time tagging (statistical, HMM)
Step 6. POS tagging
Step 7. Base phrase chunking
Step 8. Third-party time tagging (rule-based)
Step 9. Feature composition
Step 10. Multi-class SVM classification
Step 11. Post-processing

4 Chinese System

In this section we describe the TERN system developed for Chinese text. The same approach outlined in the preceding section is followed to obtain a word-level representation using the bracketing scheme. As mentioned earlier, this representation requires a sentence segmenter and a tokenizer for a given raw document that contains several sentences. Similar to the English system, the Chinese TERN data is also segmented into sentences and tokenized into words. We train a sentence segmenter and tokenizer for Chinese using the Chinese Tree Bank (CTB). The performance of the sentence segmenter is in the high 90's while the performance of the tokenizer is in the mid 90's.

We define a number of features for each token; the token itself, its part of speech, n-gram suffixes and prefixes, pattern flags and the time tag from a rule-based (hand generated) Chinese time expression tagger that we have developed. We used a part-of-speech (POS) tagger, trained in-house, to determine the POS tag for each word. This tagger is based on the Yamcha SVM toolkit and trained on the CTB. The token pattern flags are determined by looking up a number of hand-generated character lists that indicate months, dates, numbers, ordinals and foreign names. In addition to those features we also use a number of previous tag decisions as features. Summarizing, the following features are used within a predefined, finite-size sliding window:

- tokens
- POS tags
- n-gram suffixes and prefixes
- token pattern flags
- rule-based time expression tags
- previous time expression decisions

We note that the feature set for the Chinese system is not as rich as the English system due to lack of NLP resources for Chinese. However, with the availability of the CTB this is changing. We have been developing a syntactic chunker and a syntactic parser for Chinese. However, at the time of evaluation, we were not able to successfully incorporate some features extracted using those systems.

A total of 8 one-versus-all SVM classifiers were trained using a polynomial kernel of degree 2. The regularization parameter of SVMs was set to C=1.0. The class labels are illustrated below:

((*, ((*), (*, (*), (*)), *, *), O

The system architecture is shown in Figure 2. It is very similar to the English system with a few components missing. As in the English system, after the classification we employ a simple post-processing algorithm to maintain the consistency of bracketing. However, we observe that the bracketing after detection was perfectly consistent. This is in contrast with our English system, which has relatively more complex time expressions than Chinese. In the following we summarize the steps taken in the Chinese system for the extraction of time expressions:

Step 1. Sentence segmentation
Step 2. Word segmentation (tokenization)
Step 3. POS tagging
Step 4. Rule-based time tagging
Step 5. Feature composition
Step 6. Multi-class SVM classification
Step 7. Post-processing

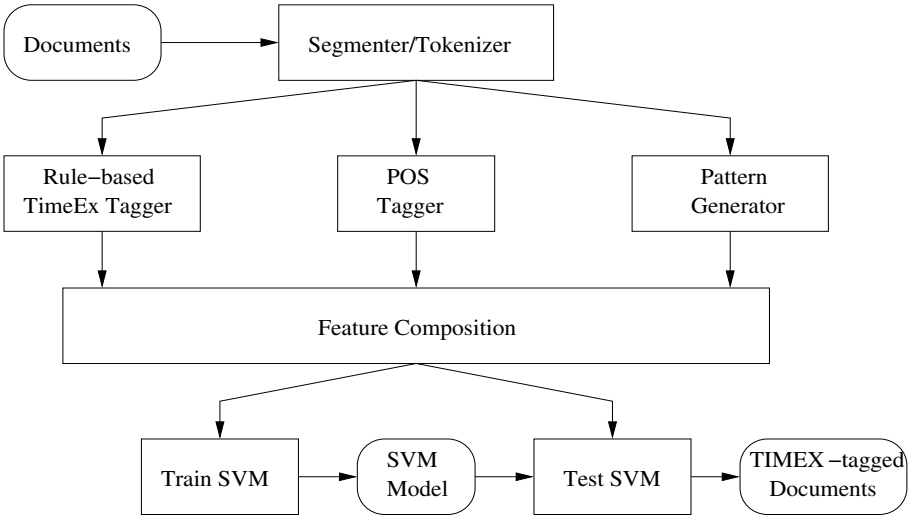

Fig. 2. General system architecture for Chinese automatic time expression labeler

5 Experimental Results

5.1 Labeling Performance

In this section we report the performance of the systems as scored against the evaluation data. We note that the system development and its optimization have been made over a development set withheld from available training data. This data was 20% and 10% of the development data for the English and Chinese systems, respectively. For evaluation, the development data was put back into the training pool and the final systems were trained using the whole training set described in Section 2.

We used the official TERN scoring script developed by MITRE for evaluating the systems. The results are reported using precision and recall numbers along with the $F_{\beta=1}$ metric. Table 2 presents the detection performance and Table 3 presents the bracketing performance as defined in the TERN 2004 Evaluation Plan [6]. In the former, the system output is counted correct if it overlaps (even if only one character) with the reference time expression. However, in the latter an exact match between the system output and the reference expressions is required. All SVM classifiers, for POS tagging, sentence segmentation, Chinese word segmentation etc. were implemented using TinySVM with a polynomial kernel of degree 2 and the general purpose SVM based chunker YamCha . In these experiments the accuracy of all those SVM-based systems were comparable to that of the state-of-the-art systems.

[6] `http://timex2.mitre.org/tern_evalplan-2004.29apr04.pdf`

Table 2. Detection Performance

	Precision	Recall	$F_{\beta=1}$
English System	97.8%	89.4%	93.5
Chinese System	96.5%	85.2%	90.5

Table 3. Bracketing Performance

	Precision	Recall	$F_{\beta=1}$
English	91.9%	84.0%	87.8
Chinese	83.8%	74.0%	78.6

Table 4. Comparison of detection and bracketing performances of word and character based Chinese systems

System	Detection	Bracketing
Word-based	90.5	78.6
Character-based	90.1	80.3

It is interesting to note that the bracketing performance (BP) of the Chinese system is considerably worse than that of the English system. This is in contrast with the detection performance (DP) which is moderately lower than the DP performance of the English system. Preliminary error analysis has shown that this is partly due to the so-called "boundary noise" at character level introduced by the Chinese word segmenter. This has motivated us to shift from using the words as tokens to using the characters as tokens for time expression labeling. We provide a result in Table 4 to show that such processing paradigm shift does indeed improve the bracketing performance with a statistically insignificant drop in detection performance. The latter is expected since the character based system has a narrower context when compared to the word-based system.

5.2 Computational Performance

In this section we provide some figures that will give a sense of how long it took for the SVM training and the final system evaluation.

For the English system, after segmentation and tokenization we had created approximately 316K distinct labeled examples for SVM training. The training took almost 10 hours. The number of binary features was 58907. For the Chinese system, we had approximately 137K distinct labeled examples. The training time was 2.5 hours. The number of binary features was 60174. Table 5 summarizes these training statistics.

Approximate processing times during evaluations for some of the system components are shown in Table 6. Both the English and Chinese systems were run on PCs with a single CPU, Pentium 4 at 2.4GHz. The machines had 1GB of RAM.

Table 5. Training statistics

	Examples	# Features	Training Time (hr)
English System	316K	58907	10
Chinese System	137K	60174	2.5

Table 6. Run times (RTs) for several components in English and Chinese systems

	English System's RT (min.)	Chinese System's RT (min.)
Segmentation/Tokenization	3	26
Part-of-Speech Tagging	34	40
Base Phrase Chunking Tagging	2	na
Dependency parsing	3	na
Rule-based Time Tagging	1	1
HMM-based Time Tagging	< 1	na
Feature Composition	< 1	< 1
SVM Time Tagging	5	3

5.3 Feature Sensitivity

In this section we explore the impact of adding or removing some of the features on performance. Computationally, it is not feasible to try all configurations; for example, we have $2^9 - 1 = 511$ possible configurations for the English system. Instead, we picked several configurations that are believed to shed light on the behavior of the system with respect to the features. The results are summarized in Table 7.

The difference in performance between the baseline system (that uses only the tokens) and the final system is 6.3% absolute in BP and 3.1% absolute in DP. This clearly indicates the significance of feature engineering. Although each feature only marginally contributes to performance, it becomes significant when all the contributions are summed up. The results show that the information provided by external time expression classifiers based on different paradigms contributed the most. It can also be seen that the syntactic features in terms of base phrase chunks did not contribute significantly. This is attributed to the fact that the temporal expressions have an unrestricted distribution with respect to syntax.

Another useful view for feature sensitivity can be obtained by grouping features into broad classes, such as

- baseline features: tokens
- lexical features: POS, lower-case, patterns, hyphen
- syntactic features: base phrase chunks
- "semantic" features: heads and grammatical relations
- external features: rule-based time tags, hmm-based time tags

and perform experiments by adding one group of features at a time. The results are presented in Table 8.

Table 7. English system performance at different feature combinations; tok: tokens, low: lower-cased tokens, pos: part-of-speech tags, bp: base phrase chunks, hyp: hyphenation flag, pat: patterns, rbtt: rule-based timex tags, iftt: identifinder timex tags, DP: detection performance and BP: bracketing performance

tok	low	pos	bp	hyp	pat	dep	rbtt	iftt	DP	BP
+	-	-	-	-	-	-	-	-	90.4	81.5
+	-	+	-	-	-	-	-	-	90.1	82.7
+	-	+	+	-	-	-	-	-	90.1	82.8
+	+	+	-	-	-	-	-	-	90.0	82.9
+	-	+	-	+	-	-	-	-	90.9	83.2
+	+	+	-	+	-	-	-	-	90.7	82.8
+	-	+	-	-	+	-	-	-	91.5	83.5
+	-	+	+	-	+	-	-	-	91.3	83.9
+	-	+	-	-	+	+	-	-	91.3	84.8
+	-	+	+	-	+	+	-	-	91.3	84.8
+	-	+	-	-	+	-	+	-	92.2	84.9
+	-	+	-	-	+	-	-	+	92.6	85.6
+	-	+	-	-	-	-	+	+	92.6	86.2
+	-	+	-	-	+	-	+	+	92.9	86.3
+	-	+	-	-	-	+	+	+	92.8	87.3
+	-	+	-	-	+	+	+	+	93.0	87.4
+	-	+	+	-	+	+	+	+	93.1	87.6
+	-	+	+	-	-	+	+	+	92.8	87.7
+	+	+	+	+	+	+	+	+	93.5	87.8

Table 8. System performance with respect to broad classes of features; lex: lexical features, syn: syntactic features, sem: "semantic" features, ext: external features, DP: detection performance and BP: bracketing performance

	DP	BP
baseline	90.4	81.5
baseline+lex	91.9	83.5
baseline+lex+syn	91.7	83.9
baseline+lex+syn+sem	91.7	85.4
baseline+lex+syn+sem+ext	93.5	87.8

Similar experiments have also been performed for the Chinese system. The results are shown in Table 9. Feature engineering contributed about 15% absolute to both detection and bracketing performances. It can be easily seen that each feature consistently contributed to the system performance. Results with different combinations clearly show the overlapping nature of the features. For example, the impact of POS tags when the other features are absent is quite different form the impact when all other features are present. A similar argument can also be made for the rule-based timex tags. The table also shows that they are the most useful features.

Table 9. Chinese system performance at different feature combinations; tok: tokens, pos: part-of-speech tags, 2-gram: prefixes and suffixes of length 2, pat: patterns, rbtt: rule-based timex tagger, DP: detection performance and BP: bracketing performance

tok	pos	2-gram	pat	rbtt	DP	BP
+	-	-	-	-	74.9	63.8
+	+	-	-	-	87.0	74.2
+	-	+	-	-	80.3	69.9
+	-	-	+	-	86.9	74.2
+	-	-	-	+	90.2	77.0
+	-	+	+	-	87.9	76.8
+	+	+	+	-	88.8	77.8
+	-	+	+	+	90.5	78.1
+	+	+	+	+	90.5	78.6

5.4 Error Analysis

We performed some preliminary error analysis without any quantification by looking at some phenomena that can be categorized at two levels; namely, token and phrase levels. At token levels, anaphoric temporal mentions were consistently missed, e.g. *this,it*, and frequent "time mentions" appearing as or part of proper nouns, or appearing as cardinal/ordinal numbers are spuriously detected, e.g. *the tonight show, midnight cowboy, 2000, first*. At phrase levels, time expressions with embedded structures are frequently detected as flat structures covering either the maximum extent or, one or multiple of its shorter mentions. This is probably due to quite small numbers of embedded time expressions in the training data. Besides, time mentions covering longer noun phrases were prematurely terminated, and, in some cases, however, shorter noun phrase time mentions were elongated by including VPs. Currently, a more detailed error analysis towards finding consistent contextual cues to avoid those phenomena in labeling is in progress.

6 Conclusions

We have introduced an end-to-end language independent statistical approach for labeling time expressions occurring in English and Chinese text. The architecture of the final systems for English and Chinese have been found very similar. The systems have had competitive performances at the evaluation workshop, although the Chinese system has not performed as well as the English system. We believe that this is partly due to the difficulty of language, partly due to the relatively smaller size of training data and partly due to the relatively small number of features employed when compared to the English system. We plan to perform detailed error analysis, use additional features (e.g. from WordNet, syntactic trees) and employ feature selection in a principled manner for further performance improvement.

Acknowledgement

We extend our special thanks to Wayne Ward, James H. Martin and Dan Jurafsky for their support, encouragement and useful feedback during this work. We also thank to Ashley Thornton for her careful and neat error analysis. This work is supported by the ARDA Aquaint II Program via contract NBCHC040040.

References

[Allwein et. al 2000] Allwein, E. L., Schapire R. E., and Singer, Y.: Reducing multiclass to binary: A unifying approach for margin classifiers. Journal of Machine Learning Research, 1:113-141, (2000).

[Bikel et. al 1999] Bikel, D. M., Schwartz, R. L., and Weischedel, R. M. , An algorithm that learns what's in a name. Machine Learning. Vol. 34, no. 1-3, pp. 211-231, (1999).

[Burges 1998] Burges, C. J. C.: Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery, 2(2), pages 1-47, (1997).

[Ferro et. al. 2004] Ferro, L., Gerber, L., Mani I., Sundheim B. and Wilson, G.: 2003 standard for the annotation of temporal expressions. Technical Report, MITRE, (2004).

[Kudo and Matsumato 2000] Kudo, T. and Matsumato, Y.: Use of support vector learning for chunk identification. Proc. of the 4th Conference on Very Large corpora, pages 142-144, (2000).

[Lin 1998] Lin D.: Dependency-based Evaluation of MINIPAR. Workshop on the Evaluation of Parsing Systems, Granada, Spain, May, (1998).

[Mani 2004] Mani, I.: Recent Developments in Temporal Information Extraction. to appear in Proceedings of RANLP'03, (2004).

[Ramhsaw and Marcus 1995] Ramhsaw, L. E. and Marcus, M. P.: Text Chunking Using Transformation Based Learning. Proceedings of the 3rd ACL Workshop on Very Large Corpora, pages 82-94, (1995).

[Reynar and Ratnaparkhi 1997] Reynar, J. C., and Ratnaparkhi, A.: A Maximum Entropy Approach to Identifying Sentence Boundaries. Proceedings of the Fifth Conference on Applied Natural Language Processing. March 31-April 3, (1997).

[Sang and Veenstra 1999] Sang, E. F. T. J. and Veenstra, J.: Representing text chunks Proceedings of EACL'99, pages 173-179, (1999).

[Saquete et. al. 2004] Saquete E., Martinez-Barco P., Munoz, R. and Vicedo J.L.: Splitting Complex Temporal Questions for Question Answering systems. Association for Computational Linguistics (ACL). Barcelona, SPAIN, (2004).

[Schilder and Habel 2003] Schilder, F. and Habel, C.: Temporal information extraction for temporal question answering. Proceedings of the 2003 AAAI Spring Symposium in New Directions in Question Answering. Stanford University, Palo Alto, USA, (2003).

[Vapnik 1995] Vapnik, V.: The Nature of Statistical Learning Theory. Springer Verlag, New York, USA, (1995).