A Comparison of Tagging Strategies for Statistical Information Extraction

Christian Siefkes
Database and Information Systems Group, Freie Universität Berlin
Berlin-Brandenburg Graduate School in Distributed Information Systems
Takustr. 9, 14195 Berlin, Germany
siefkes@mi.fu-berlin.de

Abstract

There are several approaches that model *information extraction* as a token classification task, using various *tagging strategies* to combine multiple tokens. We describe the tagging strategies that can be found in the literature and evaluate their relative performances. We also introduce a new strategy, called *Begin/After tagging* or *BIA*, and show that it is competitive to the best other strategies.

1 Introduction

The purpose of *information extraction* (IE) is to find desired pieces of information in natural language texts and store them in a form that is suitable for automatic querying and processing. IE requires a predefined output representation (*target structure*) and only searches for facts that fit this representation. Simple target structures define just a number of *slots* to be filled with a string extracted from a text (*slot filler*). For this simple kind of information extraction, statistical approaches that model IE as a *token classification* task have proved very successful. These systems split a text into a series of tokens and invoke a trainable classifier to decide for each token whether or not it is part of a slot filler of a certain type. To re-assemble the classified tokens into multi-token slot fillers, various *tagging strategies* can be used.

So far, each classification-based IE approach combines a specific tagging strategy with a specific classification algorithm and specific other parameter settings, making it hard to detect how each of these choices influences the results. To allow systematic research into these choices, we have designed a generalized IE system that allows utilizing any tagging strategy with any classification algorithm. This makes it possible to compare strategies or algorithms in an identical setting. In this paper, we describe the tagging strategies that can be found in the literature and evaluate them in the context of our framework. We also introduce a new strategy, called *Begin/After tagging* or *BIA*, and show that it is competitive to the best other strategies. While there are various approaches that employ a classification algorithm with one of the tagging strategies described below, there are no other comparative analyses of tagging strategies yet, to the best of our knowledge.

In the next section, we describe how IE can be modeled as a token classification task and explain the tagging strategies that can be used for this purpose. In Sec. 3 we describe the IE framework and the experimental setup used for comparing the various tagging strategies. In Sec. 4 we list and analyze the results of the comparison.

2 Modeling Information Extraction as a Token Classification Task

There are multiple approaches that model IE as a token classification task, employing standard

Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL, pages 149–152,
New York, June 2006.

Strategy	Triv	IOB2	IOB1	BIE	BIA	BE
Special class for first token	–	+	(+)[a]	+	+	+
Special class for last token	–	–	–	+	–	+
Special class for token after last	–	–	–	–	+	–
Number of classes	$n+1$	$2n+1$	$2n+1$	$4n+1$	$3n+1$	$2\times(n+1)$
Number of classifiers	1	1	1	1	1	2

[a] Only if required for disambiguation

Table 1: Properties of Tagging Strategies

classification algorithms. These systems split a text into a series of tokens and invoke a trainable classifier to decide for each token whether or not it is part of a slot filler of a certain type. To reassemble the classified tokens into multi-token slot fillers, various *tagging strategies* can be used.

The trivial (*Triv*) strategy would be to use a single class for each slot type and an additional "O" class for all other tokens. However, this causes problems if two entities of the same type immediately follow each other, e.g. if the names of two *speakers* are separated by a linebreak only. In such a case, both names would be collapsed into a single entity, since the trivial strategy lacks a way to mark the begin of the second entity.

For this reason (as well as for improved classification accuracy), various more complex strategies are employed that use distinct classes to mark the first and/or last token of a slot filler. The two variations of *IOB* tagging are probably most common: the variant usually called *IOB2* classifies each token as the begin of a slot filler of a certain type (B-*type*), as a continuation of the previously started slot filler, if any (I-*type*), or as not belonging to any slot filler (O). The *IOB1* strategy differs from *IOB2* in using B-*type* only if necessary to avoid ambiguity (i.e. if two same-type entities immediately follow each other); otherwise I-*type* is used even at the beginning of slot fillers. While the *Triv* strategy uses only $n+1$ classes for n slot types, *IOB* tagging requires $2n+1$ classes.

BIE tagging differs from *IOB* in using an additional class for the last token of each slot filler. One class is used for the first token of a slot filler (B-*type*), one for inner tokens (I-*type*) and another one for the last token (E-*type*). A fourth class BE-*type* is used to mark slot fillers consisting of a single token (which is thus both begin and end). *BIE* requires $4n+1$ classes.

A disadvantage of the *BIE* strategy is the high number of classes it uses (twice as many as *IOB1|2*). This can be addressed by introducing a new strategy, *BIA* (or *Begin/After* tagging). Instead of using a separate class for the last token of a slot filler, *BIA* marks the first token *after* a slot filler as A-*type* (unless it is the begin of a new slot filler). Begin (B-*type*) and continuation (I-*type*) of slot fillers are marked in the same way as by *IOB2*. *BIA* requires $3n+1$ classes, n less than *BIE* since no special treatment of single-token slot fillers is necessary.

The strategies discussed so far require only a single classification decision for each token. Another option is to use two separate classifiers, one for finding the begin and another one for finding the end of slot fillers. *Begin/End* (*BE*) tagging requires $n+1$ classes for each of the two classifiers (B-*type* + O for the first, E-*type* + O for the second). In this case, there is no distinction between inner and outer (other) tokens. Complete slot fillers are found by combining the most suitable begin/end pairs of the same type, e.g. by taking the length distribution of slots into account. Table 1 lists the properties of all strategies side by side.

3 Classification Algorithm and Experimental Setup

Our generalized IE system allows employing any classification algorithm with any tagging strategy and any context representation, provided that a suitable implementation or adapter exists. For this paper, we have used the *Winnow* (Littlestone, 1988) classification algorithm and

Strategy	IOB2	IOB1	Triv	BIE	BIA	BE
Seminar Announcements						
etime	97.1	92.4	92.0	94.4	**97.3**	93.6
location	81.7	**81.9**	81.6	77.8	**81.9**	82.3
speaker	85.4	82.0	82.0	84.2	**86.1**	83.7
stime	**99.3**	97.9	97.7	98.6	**99.3**	99.0
Corporate Acquisitions						
acqabr	55.0	53.8	53.9	48.3	**55.2**	50.2
acqloc	27.4	**29.3**	**29.3**	15.7	27.4	18.0
acquired	53.5	**55.7**	55.5	54.8	53.6	53.7
dlramt	71.7	71.5	**71.9**	71.0	71.7	70.5
purchabr	**58.1**	56.1	57.0	47.3	58.0	51.8
purchaser	55.7	55.3	**56.2**	52.7	55.7	55.5
seller	31.8	32.7	**34.7**	27.3	30.1	32.5
sellerabr	25.8	28.0	**28.9**	16.8	24.4	21.4
status	56.9	**57.4**	56.8	56.1	**57.4**	55.2

Table 2: F Percentages for Batch Training

the context representation described in (Siefkes, 2005), varying only the tagging strategy. An advantage of Winnow is its supporting *incremental* training as well as *batch* training. For many "real-life" applications, automatic extractions will be checked and corrected by a human revisor, as automatically extracted data will always contain errors and gaps that can be detected by human judgment only. This correction process continually provides additional training data, but the usual batch-trainable algorithms are not very suited to integrate new data, since full retraining takes a long time.

We have compared the described tagging strategies on two corpora that are used very often to evaluate IE systems, *CMU Seminar Announcements* and *Corporate Acquisitions*.[1] For both corpora, we used the standard setup: 50/50 training/evaluation split, averaging results over five (Seminar) or ten (Acquisitions) random splits, "one answer per slot" (cf. Lavelli et al. (2004)). Extraction results are evaluated in the usual way by calculating *precision* P and *recall* R of the extracted slot fillers and combining them in the *F-measure*, the harmonic mean of precision and recall: $F = \frac{2 \times P \times R}{P+R}$.[2] For significance testing, we applied a paired two-tailed Student's T-test on the F-measure results, without assuming the variance of the two samples to be equal.

[1]Both available from the *RISE Repository* <http://www.isi.edu/info-agents/RISE/>.

[2]This is more appropriate than measuring raw token classification accuracy due to the very unbalanced class distribution among tokens. In the *Seminar Announcements* corpus, our tokenization schema yields 139,021 tokens, only 9820 of which are part of slot fillers. Thus most strategies could already reach an accuracy of 93% by always predicting the O class. Also, correctly extracting slot fillers is the goal of IE—a higher token classification accuracy won't be of any use if information extraction performance suffers.

Strategy	IOB1	Triv	BIE	BIA	BE
etime	o (81.6%, –)	o (85.3%, –)	– (98.4%, –)	o (68.6%, +)	o (90.6%, –)
location	o (84.3%, –)	o (90.5%, –)	– (98.9%, –)	o (55.8%, +)	– (98.7%, –)
speaker	– (98.1%, –)	– (95.3%, –)	o (46.7%, –)	o (1.4%, –)	o (20.8%, –)
stime	o (92.9%, –)	– (96.9%, –)	o (75.9%, –)	o (0.0%, =)	o (85.4%, –)
acqabr	o (19.8%, –)	o (12.7%, +)	– (98.8%, –)	o (2.2%, +)	– (99.4%, –)
acqloc	o (75.0%, –)	o (77.8%, –)	– (98.1%, –)	o (11.2%, –)	– (99.3%, –)
acquired	o (17.7%, +)	o (33.6%, +)	o (9.0%, –)	o (0.3%, –)	o (8.9%, +)
dlramt	o (6.6%, –)	o (6.5%, –)	o (5.3%, –)	o (2.9%, –)	o (15.1%, +)
purchabr	o (45.1%, –)	o (37.8%, –)	– (99.9%, –)	o (14.7%, +)	o (94.0%, –)
purchaser	o (62.1%, –)	o (54.8%, –)	o (87.3%, –)	o (6.6%, –)	o (33.8%, –)
seller	o (64.3%, +)	o (72.1%, +)	o (20.1%, –)	o (2.8%, –)	o (24.6%, –)
sellerabr	o (68.0%, +)	o (64.9%, +)	o (91.9%, –)	o (0.8%, –)	o (45.2%, –)
status	o (68.8%, –)	o (70.7%, –)	o (71.7%, –)	o (18.5%, +)	o (64.7%, –)

Table 3: Incremental Training: Significance of Changes Compared to *IOB2*

Strategy	IOB1	Triv	BIE	BIA	BE
etime	o (87.3%, –)	o (91.8%, –)	o (95.0%, –)	o (18.5%, +)	– (96.9%, –)
location	o (18.8%, +)	o (0.5%, –)	– (98.9%, –)	o (22.4%, +)	o (50.3%, +)
speaker	– (98.0%, –)	– (99.1%, –)	o (67.0%, –)	o (55.2%, +)	o (88.8%, –)
stime	o (82.9%, –)	o (84.4%, –)	o (82.2%, –)	o (11.5%, –)	o (73.4%, –)
acqabr	o (49.7%, –)	o (45.8%, –)	– (99.7%, –)	o (6.8%, +)	– (97.9%, –)
acqloc	o (56.3%, +)	o (54.0%, +)	– (99.9%, –)	o (1.1%, +)	– (99.4%, –)
acquired	o (91.5%, +)	o (84.8%, +)	o (67.9%, +)	o (3.5%, +)	o (8.4%, +)
dlramt	o (5.7%, –)	o (14.3%, +)	o (30.2%, –)	o (3.3%, +)	o (46.9%, –)
purchabr	o (77.1%, –)	o (44.0%, –)	– (100.0%, –)	o (6.6%, –)	– (99.5%, –)
purchaser	o (24.1%, –)	o (26.3%, +)	– (96.0%, –)	o (2.5%, –)	o (17.5%, –)
seller	o (34.8%, +)	o (83.5%, +)	– (96.2%, –)	o (59.2%, –)	o (36.1%, +)
sellerabr	o (66.7%, +)	o (76.1%, +)	– (99.7%, –)	o (40.7%, –)	o (90.7%, –)
status	o (26.3%, +)	o (1.5%, –)	o (43.2%, –)	o (28.0%, +)	o (76.0%, –)

Table 4: Batch Training: Significance of Changes Compared to *IOB2*

4 Comparison Results

Table 2 list the F-measure results (in percent) reached for both corpora using batch training. Incremental results have been omitted due to lack of space—they are generally slightly worse than batch results, but in many cases the difference is small. For the *Corporate Acquisitions*, the batch results of the best strategies (IOB2 and BIA) are better than any other published results we are aware of; for the *Seminar Announcements*, they are only beaten by the *ELIE* system (Finn and Kushmerick, 2004).[3]

Tables 3 and 4 analyze the performance of each tagging strategy for both training regimes,

[3]cf. (Siefkes and Siniakov, 2005, Sec. 6.5)

using the popular *IOB2* strategy as a baseline. The first item in each cell indicates whether the strategy performs significantly better ("+") or worse ("–") than *IOB2* or whether the performance difference is not significant at the 95% level ("o"). In brackets, we show the significance of the comparison and whether the results are better or worse when significance is ignored.

Considering these results, we see that the *IOB2* and *BIA* strategies are best. No strategy is able to significantly beat the *IOB2* strategy on any slot, neither with incremental nor batch training. The newly introduced *BIA* strategy is the only one that is able to compete with *IOB2* on all slots. The *IOB1* and *Triv* strategies come close, being significantly worse than *IOB2* only for one or two slots. The two-classifier *BE* strategy is weaker, being significantly outperformed on three (incremental) or four (batch) slots. Worst results are reached by the *BIE* strategy, where the difference is significant in about half of all cases. The good performance of *BIA* is interesting, since this strategy is new and has never been used before (to our knowledge). The *Triv* strategy would have supposed to be weaker, considering how simple this strategy is.

5 Conclusion

Previously, classification-based approaches to IE have combined a specific tagging strategy with a specific classification algorithm and specific other parameter settings, making it hard to detect how each of these choices influences the results. We have designed a generalized IE system that allows exploring each of these choices in isolation. For this paper, we have tested the tagging strategies that can be found in the literature. We have also introduced a new tagging strategy, *BIA* (*Begin/After* tagging).

Our results indicate that the choice of a tagging strategy, while not crucial, should not be neglected when implementing a statistical IE system. The *IOB2* strategy, which is very popular, having been used in public challenges such as those of *CoNLL* (Tjong Kim Sang and De Meulder, 2003) and *JNLPBA* (Kim et al., 2004), has been found to be indeed the best of all established tagging strategies. It is rivaled by the new *BIA* strategy. In typical situations, using one of those strategies should be a good choice—since *BIA* requires more classes, it makes sense to prefer *IOB2* when in doubt.

Considering that it is not much worse, the *Triv* strategy which requires only a single class per slot type might be useful in situations where the number of available classes is limited or the space or time overhead of additional classes is high. The two-classifier *BE* strategy is still interesting if used as part of a more refined approach, as done by the *ELIE* system (Finn and Kushmerick, 2004).[4] Future work will be to observe how well these results generalize in the context of other classifiers and other corpora. To combine the strengths of different tagging strategies, ensemble meta-strategies utilizing the results of multiple strategies could be explored.

References

Aidan Finn and Nicholas Kushmerick. 2004. Multi-level boundary classification for information extraction. In *ECML 2004*, pages 111–122.

Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. 2004. Introduction to the bio-entity recognition task at JNLPBA. In *BioNLP/NLPBA 2004*.

A. Lavelli, M. Califf, F. Ciravegna, D. Freitag, C. Giuliano, N. Kushmerick, and L. Romano. 2004. A critical survey of the methodology for IE evaluation. In *LREC*.

Nick Littlestone. 1988. Learning quickly when irrelevant attributes abound. *Machine Learning*, 2.

Christian Siefkes and Peter Siniakov. 2005. An overview and classification of adaptive approaches to information extraction. *Journal on Data Semantics*, IV:172–212. LNCS 3730.

Christian Siefkes. 2005. Incremental information extraction using tree-based context representations. In *CICLing 2005*, LNCS 3406. Springer.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *CoNLL-2003*.

[4] They augment the *BE* strategy with a second level of begin/end classifiers for finding suitable tags matching left-over tags from the level-1 classifiers.