Bootstrapping Noun Groups Using Closed-Class Elements Only

Kathrin Eichler
DFKI - Language Technology
Berlin, Germany
kathrin.eichler@dfki.de

Günter Neumann
DFKI - Language Technology
Saarbrücken, Germany
neumann@dfki.de

Abstract

The identification of noun groups in text is a well researched task and serves as a pre-step for other natural language processing tasks, such as the extraction of keyphrases or technical terms. We present a first version of a noun group chunker that, given an unannotated text corpus, adapts itself to the domain at hand in an unsupervised way. Our approach is inspired by findings from cognitive linguistics, in particular the division of language into open-class elements and closed-class elements. Our system extracts noun groups using lists of closed-class elements and one linguistically inspired seed extraction rule for each open class. Supplied with raw text, the system creates an initial validation set for each open class based on the seed rules and applies a bootstrapping procedure to mutually expand the set of extraction rules and the validation sets. Possibly domain-dependent information about open-class elements, as for example provided by a part-of speech lexicon, is not used by the system in order to ensure the domain-independency of the approach. Instead, the system adapts itself automatically to the domain of the input text by bootstrapping domain-specific validation lists. An evaluation of our system on the Wall Street Journal training corpus used for the CONLL 2000 shared task on chunking shows that our bootstrapping approach can be successfully applied to the task of noun group chunking.

1 Introduction

The identification of noun groups (or chunks) in text is a well researched task and serves as a pre-step for other natural language processing tasks, e.g. the extraction of keyphrases or technical terms as for example in [Eichler and Neumann, 2010]. Our approach to identifying noun groups in text is inspired by findings from cognitive linguistics, in particular the division of concepts expressed in language into two subsystems: the grammatical subsystem and the lexical subsystem [Talmy, 2000].[1] The lexical subsystem is expressed using so-called open-class elements (OCEs), i.e. nouns, verbs, adjectives and adverbs. Concepts associated with the grammatical subsystem are expressed using so-called closed-class elements (CCEs), including function words such as conjunctions, determiners, pronouns, and prepositions, but also suffixes such as plural markers and tense markers. Consider the following example, taken from [Evans and Pourcel, 2009], with CCEs printed in bold:

> ***A*** *waiter serv****ed the*** *customer****s***

CCEs exhibit two important characteristics: First, *the inventory of CCEs is fixed*, i.e., whereas OCEs are constantly added to the language and vary enormously depending on the domain of the input text, the set of CCEs is limited, does not change over time and is the same for all domains. Due to the limited number of CCEs, finite CCE lists can be generated with fairly little effort for basically any language. Second, *CCEs occur very frequently*[2] *and provide a structuring function*, i.e. a 'scaffolding', across which concepts associated with the lexical subsystem can be draped [Evans and Pourcel, 2009].

We present a first version of a noun group chunker that makes use of this structuring function of CCEs. The general idea is to provide domain-independent information only (i.e. the CCE lists and a few linguistically-inspired seed rules) and make the system adapt itself automatically to the domain of the input text by bootstrapping domain-specific validation lists.

Based on the lists of CCEs and one seed extraction rule for each of the four OCE classes noun (N), verb (V), adjective (ADJ) and adverb (ADV), the system creates an initial validation set for each OCE class. A bootstrapping procedure is used to mutually expand the set of extraction rules and the validation sets in order to eventually assign one of the four OCE tags to all unknown (i.e. non-CCE) tokens in the input text. Based on the final tagging, sequences of ADJ and N tokens are extracted as noun groups.

The algorithm is described in detail in section 3. Evaluation results are presented in section 4.

2 Related Work

As our approach towards noun group extraction is based on the assignment of word class tags, our work is related to the task of part-of speech (POS) tagging. Unsupervised approaches to POS tagging usually disambiguate tags using a lexicon of possible tags for each token (e.g. [Merialdo, 1994], [Goldwater and Griffiths, 2007] and many others). However, these lexicons are large and, due to the openness of the OCE classes, can never be exhaustive. [Haghighi and Klein, 2006] replace the tagging lexicon by a prototype list, specifying three examples of each tag. We reduce the

[1] Note that Talmy considers this linguistic structuring as universal, i.e., it holds for any specific natural language. Hence, our approach reveals a high degree of language independency.

[2] Closed-class words constitute about 40% of an average English text [Höhle and Weissenborn, 1999].

used lexicon to the possible tags of CCEs to ensure domain-independency.

Our bootstrapping algorithm is similar to the procedures described by [Riloff and Jones, 1999] and [Collins and Singer, 1999] for labelling words with semantic categories. Starting with a small set of seed words representing each target semantic category and an unannotated corpus, [Riloff and Jones, 1999] create extraction patterns using syntactic templates, compute a score for each pattern based on the number of seed words among its extractions and use the best patterns to automatically label more words. Each newly labeled word is assigned a score based on how many different patterns extracted it and the best words remain in the semantic dictionary on which the next bootstrapping iteration is based. [Collins and Singer, 1999] use spelling rules in addition to contextual rules. [Yangarber *et al.*, 2002] present a bootstrapping approach to simultaneously learn diseases and locations and stress the usefulness of competing target categories. Similarly, we simultaneously learn extraction rules for all four OCE types.

3 Algorithm

3.1 CCE lists

Our CCE lexicon is based on the lists generated by [Spurk, 2006], to which we made some minor modifications. For example, we added a list of ordinal numbers (i.e. *first*, *second*, etc.), and introduced a special tag for the negation *not*. We also added a list of quantifiers used for grading, i.e *more*, *most*, *less*, *least*. These are used as part of the seed rule for extracting adjectives. We also removed Spurk's list of adverbs, which are strictly speaking OCEs.

3.2 General algorithm

The algorithm can be subdivided into four parts:

1. **Initialization**: Extract the initial validation sets based on the seed rules.
2. **Bootstrapping**: Iteratively expand the validation sets and the set of extraction rules and retag the input text.
3. **Postprocessing**: Tag all ambiguous and untagged tokens.
4. **Noun group extraction**: Extract noun groups based on the tagging.

Each of these parts is described in detail in the following sections.

3.3 Initialization

To initialize the bootstrapping process, we manually specified one seed rule for each of the four OCE types. The rules are listed in Table 1, where X represents any single non-CCE token, DET a determiner, PREP a preposition, PUNCT a punctuation symbol, BE some form of the auxiliary verb *be* and GRAD_ADV one of the four grading quantifier listed in section 3.1. The seed rule for adverbs makes use of the bound CCE *-ly*, with which adverbs are generated from adjectives. Based on the set of adjectives extracted using the adjective seed rule, we find adverbs by matching adjective seeds followed by *-ly*. Each rule extracts the part in bold as instance of the respective OCE type. It is assumed that these seed rules are trustworthy in the sense that the found matching elements for X are considered correct.

OCE type	*Seed rule*	*Example*
N	DET **X** PREP	the **computation** of
V	*to* **X** DET	to **give** the
ADJ	BE GRAD_ADV **X** PUNCT	is very **proud** .
ADV	**ADJ-ly**	**proudly**

Table 1: Seed rules

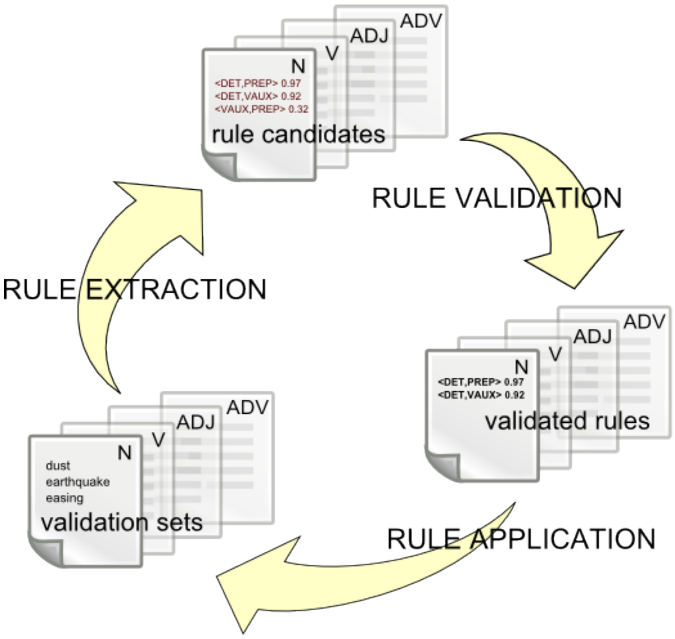

Figure 1: Bootstrapping loop

3.4 Bootstrapping

The bootstrapping loop is depicted in Figure 1. The three steps of each iteration are described in the following. The bootstrapping ends when no more rules are learnt.

Step 1: Rule Application

In the first iteration, we set the set of validated rules equal to the set of seed rules described in section 3.3. In the rule application step, the validated rules are applied to the input text by extracting all instances of X matching the respective pattern. For example, the seed rule <DET X PREP> for nouns extracts the seed *airport* from the sentence

> *Getting to and from <the:DET airport:X in:PREP> coming weeks may be the problem, however.*

The extracted instances are added to the validation set of the respective OCE class. For each extracted verb form, we also add other verb forms matched in the text, which are automatically generated based on the bound CCEs for verbs (i.e. *-(e)d*, *-s*, and *-ing*). For extracted adjectives, we also add the adverb built using *-ly* if found in the text.

After all rules have been applied, the input text is retagged based on the expanded validation sets. Tokens appearing in more than one validation list are assigned all possible tags, i.e. left ambiguous.

Step 2: Rule Extraction

New rule candidates are extracted using the validation sets generated in step 1. For each OCE class O, the method below is applied:

1. For each entry E in the validation set of O, match all <POS_L E POS_R> in the text, where POS_L (POS_R) correspond to the POS tag of the left (right) context of E, i.e. an already tagged token directly preceding (following) E.
2. Add <POS_L X POS_R> to the set of rule candidates for O.

Note that in the first iteration POS_L and POS_R represent some CCE tag. In later iterations, with some OCE tokens tagged based on the validation lists, POS_L and POS_R can also refer to an OCE tag.

For illustration of the rule extraxtion procedure, consider the following example. For the entry *airport* from the validation set of nouns, we can extract the rule candidate <DET X VAUX> from the sentence

> *While <the:DET airport:N was:VAUX> closed , flights were diverted.*

This rule is then put to the set of rule candidates for nouns.

Step 3: Rule Validation

The rule candidates extracted in step 2 are validated by calculating the accuracy of each rule candidate *r* for OCE *O* using formula 1,

$$acc(r) = \frac{pos_r + 1}{pos_r + neg_r + 1} \quad (1)$$

where pos_r refers to the number of occurrences matching the pattern <POS_L X_O POS_R>, neg_r refers to the number of occurrences matching the pattern <POS_L $X_{\neg O}$ POS_R>, and 1 is a smoothing constant. X_O refers to any token tagged with category O, $X_{\neg O}$ refers to any token tagged with any open class other than O.

If the calculated accuracy of a rule candidate exceeds a fixed threshold (currently set to 0.5), the rule candidate is added to the set of validated rules.

The input text is retagged based on the validated rules and again, ambiguous tokens are tagged with all possible tags.

3.5 Postprocessing

The postprocessing step serves two purposes: First, disambiguate all OCE tokens tagged with more than one tag. Second, tag all those tokens not appearing in any of the validation sets and not covered by any of the learned rules. To disambiguate OCE tokens to which more than one tag has been assigned, we compare the scores of all rules matching the context of the token and apply the highest-scoring one. Several matching rules are possible if the context of the token contains ambiguous tokens. For example, in order to tag the unknown token *August* in the sentence

> *Trade figures fail to show a substantial improvement from July <and:CONJ August:X 's:DET/VAUX> near-record deficits*,

we need to decide whether to apply the rule *<CONJ X DET>* (a verb rule) or *<CONJ X VAUX>* (a noun rule). As *score(<CONJ X VAUX>)* $= 0.94$ and *score(<CONJ X DET>)* $= 0.80$, we decide to apply the higher scoring noun rule and tag *August* as N.

To tag tokens that do not match any of the rules, we apply a backup procedure: We tag it based on its left context only. Here, we collect all rules with a matching left context, compute the average score of all rules for each of the tags in question, and assign the tag with the highest average score.

The postprocessing step is iterated until all tokens have been assigned a single tag.

3.6 Noun group extraction

After all tokens have been tagged, noun groups are collected by extracting all sequences of consecutive ADJ and N tokens. Note that all previous steps consider single tokens, not token sequences, i.e. a learned rule cannot be applied to extract a multi-word noun group directly. Instead, multi-word noun groups are extracted by learning and applying rules that involve OCE tags, e.g. the learned rule <DET X N> for tagging nouns, which tags *U.K.* as N in the sentence

> *But consumer expenditure data released Friday don't suggest that <the:DET U.K.:X economy:N> is slowing that quickly,*

given that *economy* has already been tagged as N.

4 Evaluation

The algorithm was evaluated on sections 15 to 18 of the Wall Street Journal corpus, a commonly used corpus for part-of speech tagging and chunking tasks, e.g. the CONLL 2000 shared task on chunking.[3] It contains 8,936 sentences with 46,874 noun groups (matching the regular expression $JJ^*(NNP|NN|NNS)^+$).

It is difficult to compare our system to others, which make use of more resources. The F-measure values of published results for the same dataset lie in the lower 90s, with a baseline F-measure of about 80 (cf. http://ifarm.nl/erikt/research/np-chunking.html). However, all these systems use a POS-annotated corpus as input, i.e., unlike our system, they require POS information to be available.

Due to the difficulty of comparison to other results, we decided to evaluate the system by taking a look at the learning process, and evaluate the effect of the initial seed rules as well as the bootstrapping and postprocessing procedures. As baseline, we tagged all non-CCE tokens with the most probable tag, N, thus extracting all chunks occurring between two CCEs as noun groups. We also evaluated the chunking result achieved using the initial validation sets extracted based on the four seed rules. Here, all tokens occurring in one of the initial validation sets were tagged accordingly, all other OCE tokens were tagged as N. In addition, we evaluated the final tagging, which was generated by applying all rules validated by the bootstrapping process as described in 3.5. The bootstrapping stopped after 7 iterations. All results are presented in Tables 2 and 3.

In the token-based evaluation, we look at the noun group tokens individually and evaluate how many of them are correctly considered part of a noun group by the algorithm. This evaluation procedure is similar to the one used for the CONLL 2000 shared task, which is also token-based. However, we do not evaluate based on the BIO tagging scheme, but count matching noun group tokens, irrespective of their position within the chunk.

The chunk-based evaluation is more strict in that it considers complete chunks only, i.e. if two of three tokens in a noun group have correctly been assigned an N tag, they are not counted as a match because one token is missing, i.e. the complete chunk was not recognized.

[3] http://www.cnts.ua.ac.be/conll2000/chunking/

	Precision	*Recall*	*F1*
Baseline	0.65	0.97	0.78
Initial tagging	0.68	0.96	0.79
Final Tagging	0.74	0.94	0.83

Table 2: Token-based evaluation of the bootstrapping

	Precision	*Recall*	*F1*
Baseline	0.50	0.66	0.57
Initial tagging	0.54	0.68	0.60
Final Tagging	0.60	0.72	0.66

Table 3: Chunk-based evaluation of the bootstrapping

5 Discussion and future work

We presented a first version of a self-adaptive noun chunker, which uses lists of closed-class elements, one seed extraction rule for each open class and a bootstrapping procedure to automatically generate and extend OCE validation sets and expand the set of extraction rules. An evaluation of the system's learning progress showed the usefulness of the bootstrapping procedure. The results are preliminary as we are presenting ongoing work, improvements are expected by optimizing the validation procedure. Currently, we only validate the rule candidates. Validating the extracted OCE tokens before adding them to the validation lists would make the algorithm more robust and prevent extraction errors from being propagated. In addition, the application of more sophisticated rule validation techniques, e.g. EM-based confidence estimation as described and used by [Jones, 2005] and [Tomita *et al.*, 2006], could improve the results.

In the current system, bound CCEs only play a minor role: When building additional verb forms for the extracted verbs and when building adverbs from adjectives. In the future, we also want to use bound CCEs to generate rules dealing with the morphology of the OCE tokens (i.e. add a second type of extraction rule, similar to the spelling features used by [Collins and Singer, 1999]).

We are currently evaluating the system on other, more specialized corpora in order to show its domain-independency. We also plan to evaluate it on texts in other languages. In addition, the influence of the size of the input text needs to be evaluated.

Acknowledgments

The work described in this paper has been carried out in the context of the research project DiLiA, co-funded by the European Regional Development Fund in the context of Investitionsbank Berlins ProFIT program (grant number: 10140159), and supported by a research grant from the German Federal Ministry of Economics and Technology to the project Theseus Ordo TechWatch (FKZ: 01MQ07016). We gratefully acknowledge this support.

References

[Collins and Singer, 1999] M. Collins and Y. Singer. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 100–111, Maryland, USA, 1999.

[Eichler and Neumann, 2010] K. Eichler and G. Neumann. DFKI KeyWE: Ranking keyphrases extracted from scientific articles. In *Proc. of the 5th International Workshop on Semantic Evaluations, ACL*, 2010.

[Evans and Pourcel, 2009] V. Evans and S. Pourcel, editors. *New Directions in Cognitive Linguistics*. Human Cognitive Processing. John Benjamins, 2009.

[Goldwater and Griffiths, 2007] S. Goldwater and T. Griffiths. A Fully Bayesian Approach to Unsupervised Part-of-Speech Tagging. In *Proc. of the ACL*, 2007.

[Haghighi and Klein, 2006] A. Haghighi and D. Klein. Prototype-Driven Learning for Sequence Models. In *Proc. of HLT/NAACL*, 2006.

[Höhle and Weissenborn, 1999] B. Höhle and J. Weissenborn. Discovering grammar. In A.D.Friederici and R. Menzel, editors, *Learning: Rule Extraction and Representation*. Walter de Gruyter, Berlin, 1999.

[Jones, 2005] Rosie Jones. *Learning to Extract Entities from Labeled and Unlabeled Texts*. PhD thesis, School of Computer Science, Carnegie Mellon University, 2005.

[Merialdo, 1994] B. Merialdo. Tagging english text with a probabilistic model. *Computational Linguistics*, 20(2):155–171, 1994.

[Riloff and Jones, 1999] E. Riloff and R. Jones. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the 16th National Conference on AI (AAAI-99)*, Orlando, FL, 1999.

[Spurk, 2006] C. Spurk. Ein minimal berwachtes Verfahren zur Erkennung generischer Eigennamen in freien Texten. Diplomarbeit, Saarland University, Germany, 2006.

[Talmy, 2000] L. Talmy. *Towards a cognitive semantics*. MIT Press, Cambridge, MA, 2000.

[Tomita *et al.*, 2006] J. Tomita, S. Soderland, and O. Etzioni. Expanding the recall of relation extraction by bootstrapping. In *EACL Workshop on Adaptive Text Extraction and Mining*, 2006.

[Yangarber *et al.*, 2002] R. Yangarber, L. Winston, and R. Grishman. Unsupervised learning of generalized names. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, 2002.