Detecting and Resolving Shell Nouns in German

Adam Roussel
Department of Linguistics
Ruhr-Universität Bochum
Bochum, Germany
roussel@linguistics.rub.de

Abstract

This paper describes the design and evaluation of a system for the automatic detection and resolution of shell nouns in German. Shell nouns are general nouns, such as *fact*, *question*, or *problem*, whose full interpretation relies on a content phrase located elsewhere in a text, which these nouns simultaneously serve to characterize and encapsulate. To accomplish this, the system uses a series of lexico-syntactic patterns in order to extract shell noun candidates and their content in parallel. Each pattern has its own classifier, which makes the final decision as to whether or not a link is to be established and the shell noun resolved. Overall, about 26.2% of the annotated shell noun instances were correctly identified by the system, and of these cases, about 72.5% are assigned the correct content phrase. Though it remains difficult to identify shell noun instances reliably (recall is accordingly low in this regard), this system usually assigns the right content to correctly classified cases.

1 Introduction

The term *shell noun* refers to the way in which particular general nouns are used to characterize and encapsulate a complex chunk of information for later reference, which might ordinarily be realized by a verb phrase or a sentence (Schmid, 2000). Example (1) below represents a typical shell noun instance.[1]

(1) Ich finde die **Tatsache**, <u>dass es keine Dinosaurier mehr gibt</u>, sehr traurig.
‘I find the fact that there are no more dinosaurs very sad.’

As this encapsulation of information coincides with an ability to link information across sentences, shell nouns are an important means of text or discourse coherence. They are also relatively common: Schmid (2000, p. 6) observes that many of the English nouns that can function as shell nouns are among the hundred most frequent nouns in the English language. However, the complete interpretation of a particular shell noun instance is only possible together with the complex content to which it, in one way or another, ‘refers’. Shell nouns must be *resolved* to be properly interpreted. Thus, the resolution of shell nouns and their content forms an essential part of any NLP system for which a degree of natural language understanding is necessary, including summarization, question answering, and sentiment analysis.

This paper will describe a system that was implemented with the aim of identifying which nouns in a given text act as shell nouns and establishing a link between these instances and the content they refer to and serve to characterize.

In contrast to previous attempts to resolve shell nouns, in which it was known which noun instances were to be considered shell nouns, the current system does not know a priori which nouns may act as shell nouns, and it does not know which of these potential shell noun instances actually require resolution. The system therefore must simultaneously decide whether a given noun instance is acting as a shell noun and resolve it to its content.

2 The Algorithm

Extraction patterns One of the most salient aspects of the phenomenon of shell nouns is their tendency to be used in certain syntactic patterns (*the fact that . . . , the question is whether . . .* , etc.), such that these patterns are sometimes used to gather shell noun instances (Schmid, 2000; Simonjetz, 2015) and to resolve them (Kolhatkar and Hirst, 2014). I use this aspect of the phenomenon as a starting point, so that this linguistic knowl-

[1] Shell nouns are in boldface, content phrases underlined.

edge ensures a basic level of functionality. The system as implemented uses an ordered sequence of nine extraction patterns, which are used to identify potential shell nouns along with their content phrase candidates. Capturing shell noun candidates together with potential content phrases has the additional benefit of reducing the range of candidates the system must consider—an important consideration, since content phrases can take on a variety of syntactic shapes and the system might otherwise be overwhelmed by candidates.

(2) Der Bildschirm geht nicht mehr an. Dieses **Problem** muss noch gelöst werden.
'The monitor won't turn on anymore. This problem must still be solved.'

The nine extraction patterns used here range from the *NN-dass* pattern, which would capture example (1), to the *PDAT-last-sent* pattern, which covers anaphoric shell nouns, i.e. cases in which the shell noun refers back to previous sentences, such as example (2) above. They are based in part on the patterns suggested in Simonjetz (2015), which are versions of Schmid's patterns adapted for use with German-language data and which have also been converted to dependency-based patterns in order to account for German's more flexible word order.

It is important to note that the patterns as used here are not intended primarily to filter noun instances or to discover shell noun instances by themselves, rather the extraction patterns serve mainly to select likely candidates for the shell noun content, based on what we know about the behavior of shell nouns. By adding more such patterns, the system can be extended to search more environments for content phrases.

Classifiers However, though there is a close association between shell nouns and particular patterns, a noun that occurs in one of these patterns is not necessarily a shell noun usage. *Grund* in example (3) occurs in the *NN-zu* pattern, but the infinitive verb phrase does not contain the shell noun's content, as the pattern predicts. And even nouns that can and often do act as shell nouns will also occur in non–shell noun usages, as in example (4), in which the noun *Entscheidung* is not being used as a shell noun, though it is capable of fulfilling this function.

Name	Description
`NN-dass`	NN with a *dass*-phrase
`NN-ist-dass`	NN and *dass*-phrase connected by a form of *sein*
`NN-KOUS`	Like `NN-dass`, for other KOUS
`NN-zu`	NN with a dependent *zu*-form verb
`NN-ist-zu`	NN and *zu*-phrase connected by a form of *sein*
`NN-PP`	NN with dependent PP
`NN-NN`	NN with dependent NP in genitive case
`PDAT-last-sent`	NN with PDAT determiner, with previous sentence root as content phrase
`PDAT-last-verb`	NN with PDAT determiner, with last verbal element as root of content phrase

Table 1: Extraction patterns used in this system

(3) Das ist für mich ein **Grund**, jetzt abzustimmen.
'That is for me a reason to vote now.'

(4) Die Entscheidung der Kommission sollte das verhindern.
'The commission's decision should prevent that.'

In order to determine which of the extracted candidate pairs constitute actual shell noun instances, I use a series of Naive Bayes classifiers,[2] which make the final decisions as to whether or not a given noun is to be regarded as a shell noun and resolved to some content phrase. Naive Bayes were chosen for this application due to their effectiveness with small amounts of training data and imbalanced training data (Müller, 2008, p. 187), both of which are the case for this dataset.

Each pattern is associated with its own classifier, such that each classifier is free to focus on the features that are most important for that particular pattern. If a classifier approves a particular pattern match, then that instance is considered a positive instance. If a pattern match is classified as a negative instance, then the system will continue and try to apply the remaining patterns, testing various candidate shell noun–content pairs. This architecture means that the order in which patterns are applied is significant, and the patterns are roughly ordered with respect to the perceived probability that they will result in matches (cf. the ordering of sieves in Lee et al. (2013)).

[2] All classifiers, including the baseline classifiers, are from the `scikit-learn` package (Pedregosa et al., 2011).

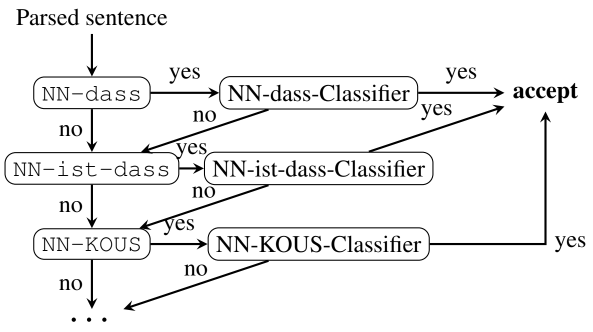

Figure 1: System architecture

Figure 1 illustrates graphically how the algorithm's various components interact.

Classification features The classifiers use a number of features[3] encompassing semantic, syntactic, and surface-level information to make its decisions regarding the status of a particular noun–content pair.

The lemma of the candidate shell noun is perhaps the most import feature, since individual shell nouns are known to prefer certain environments and disprefer others. *Tatsache* 'fact' is likely to occur often with *dass* 'that' clauses, since such clauses are associated with propositions and facts, to some degree, can be thought of as propositions that are true. *Frage* 'question', on the other hand, could conceivably be associated with *ob* 'whether' clauses.

In order to help recognize novel shell nouns and to operationalize some degree of 'abstractness', the system leverages the lexical database GermaNet (Hamp and Feldweg, 1997). Lexemes in the GermaNet database are organized hierarchically according to hypo-/hyperonomy relations, and at the top-level are a small number of semantic fields. On the hypothesis that certain of these semantic fields (e.g., 'cognition', 'communication') could correlate with whether or not a particular noun lemma might act as a shell noun, I include a noun's semantic field as a feature. GermaNet also includes subcategorization information, so I also include features indicating whether or not the verb, of which a particular noun is an argument, may ordinarily accept verbal or clausal complements (this being a rough approximation of Eckert and Strube (2000)'s I-incompatibility constraint).

The system also includes a number of syntactic or relational features, such as are also used in Müller (2007) and Jauhar et al. (2015). These include such things as the distance between the shell noun and the root node of its content phrase, the grammatical functions of shell noun and content and whether these match, and the type of determiner used with the candidate shell noun.

Finally, in order to help the system recognize nominalized content phrases (e.g. *die* ***Möglichkeit*** *der Aktualisierung der Software* 'the opportunity to update the software'), which are especially important for German-language data, I include a number of surface-level features, such as whether or not a lemma ends with *-ung*, *-keit*, or *-heit*, since these endings are typically associated with nominalized verbs or with more 'abstract' entities.

3 Data

I evaluated this approach using the German-language data from the German/English Parallel Shell Noun Corpus (Simonjetz and Roussel, 2016).[4] This corpus includes manually annotated shell noun complexes in 371 speaker turns from the Europarl corpus, which have been automatically tagged and parsed using Mate tools (Bohnet et al., 2013). The annotators in that study annotated shell nouns according to three main attributes:

1. "incompleteness": Shell nouns possess a semantic gap that is to be filled by a content phrase, which the shell noun also serves to characterize and describe.
2. "reference": Shell nouns refer to some content that occurs somewhere else in a discourse.
3. "abstractness": Shell nouns refer to entities, which are abstract and complex, such as facts, states-of-affairs, or propositions.

In the German-language data used here there are 1086 annotated noun instances, of which 466 are shell noun usages. Due to the small amount of manually annotated data available, all of the subsequent experiments described here have been performed using 5-fold cross validation.

Since only 50 nouns are completely annotated in this dataset, there remain a large number of nouns whose status is unclear. Disregarding these cases entirely would unfairly favor the baselines, which produce a large number of what are almost

[3]See Table 4 in the appendix for a summary of all of the features used here.

[4]Available at `https://github.com/ajroussel/shell-nouns-data`.

certainly false positives. However, always counting these as false positives would also be unfair, since some proportion are certainly actual shell noun instances. Therefore, in the following evaluation, I will assume that 61% of these unannotated cases are false positives, since this is the proportion of negative instances for the nouns in the corpus that are annotated.

4 Evaluation

In order to better understand the performance of this system, I will employ two baseline systems. The Constant baseline uses a classifier that always accepts any shell noun candidate that matches some pattern. It gives us an idea of what the maximum recall could be, given the current pattern set, and it also shows how far we can get using patterns alone. The Stratified baseline approves, at random, a number of candidate cases proportional to the frequency of positive instances in its training data. This baseline gives us an idea of the maximum precision and accuracy we can expect to achieve simply by choosing fewer positive cases.

Here I measure two main aspects of the system's performance: (1) To what degree are the noun instances classified correctly (regardless of the content assigned)? (2) Of the instances that are correctly classified, how many are also assigned the correct content phrase? Since the first question concerns classification performance, I use precision, recall, and F_1 score to answer this question. As for the second question, since the patterns always suggest a content phrase and this can only be correct or incorrect, I only measure accuracy with respect to content phrases ("Res" in Tables 2 and 3).

The performance of the Constant baseline shows that the patterns alone cover only about half of the cases in the test data. This system correctly classifies about half of the instances matched by some pattern, resulting in a recall of 24.2%. At the same time, the Naive Bayes classifier allows the system to produce significantly fewer false positives, and the resulting precision 56.4% is a significant improvement over both baselines.

The improvement in the system's accuracy over the baseline (72.5% vs. 57.7%) also shows that the correct pattern classifiers, rather than simply the first matching patterns, tend to approve each instance, suggesting that the system has some ability to handle such confusing cases as in example (3).

Name	P	R	F_1	Res
Constant	0.072	**0.494**	0.125	0.577
Stratified	0.178	0.060	0.090	**0.820**
This system	**0.564**	0.262	**0.356**	0.725

Table 2: Performance of the shell noun resolution algorithm (Res = resolution accuracy).

Name	P	R	F_1	Res
All	0.559	**0.277**	**0.367**	0.700
No lemmas	0.394	0.185	0.250	0.792
No GermaNet	0.682	0.251	0.366	0.736
Only lemmas	**0.741**	0.163	0.264	**0.799**

Table 3: Comparing various feature sets.

5 Related Work

Müller (2007) and Jauhar et al. (2015) attempt to automatically resolve instances of discourse deixis, specifically the anaphors *this*, *that*, and *it*, to their verbal antecedents. Using a maximum entropy classifier and a series of morphological and syntactic features, as well as some corpus-based features based on Eckert and Strube (2000)'s compatibility constraints, their algorithm achieves an F_1 score of 12.59 (P = 13.42, R = 11.84) for VP antecedents.

Jauhar et al. (2015) separates this task into two discrete stages, using a different classifier and different features for each stage. In the first stage 'classification', the classifier decides whether or not a particular pronominal instance refers to some verbal instance and thus requires resolution, and in the second stage 'resolution', their system selects the highest-scoring antecedent for this pronominal instance. For the classification stage their system has an F_1 score of 38.6 (P = 35.2, R = 42.9) and for the resolution stage, using the system classifications, 22.2 (P = 22.6, R = 21.8).

Kolhatkar and Hirst (2012) and Kolhatkar et al. (2013) use an SVM ranking algorithm to resolve instances of six anaphoric shell nouns, i.e. cases which refer back to content in previous sentences. The authors include a number of features similar to those used in Müller (2007) and Jauhar et al. (2015), such as antecedent length, syntactic type, and distance in tokens, as well as a few that are specific to the behavior of *issue* as a shell noun: use with a *whether* clause, antecedent is a question, etc. These systems had accuracies ranging from 35% to 72%, depending on the shell noun.

Kolhatkar and Hirst (2014) resolved instances of 12 English shell nouns using lexico-syntactic patterns and linguistic cues, as suggested by Schmid (2000), with the primarily goal of using this linguistic information to improve the resolution of shell nouns over the use of patterns alone. Their results (accuracy between 62% and 83%) show that this information is more useful for particular nouns whose requirements are very specific.

In order to resolve abstract anaphora (including shell nouns) Marasović et al. (2017) apply a Siamese LSTM neural network to the context of an anaphor and an antecedent candidate, thus training the network to recognize compatible anaphors and antecedents. The resulting model resolves 76.09% to 93.14% of the shell nouns in the Kolhatkar et al. (2013) dataset to the correct antecedent. On the ARRAU dataset (Poesio et al., 2013), for which Marasović et al. automatically generate training data, their system still resolves 51.89% of annotated shell noun instances correctly.

6 Discussion and Future Work

Though, in general, the implemented system has relatively high precision for the task and tends to link the identified shell nouns to the correct content, recall remains low. There are a number of potential explanations for this, each of which suggests a path along which future work could proceed.

Some of the errors the system produces appear to be related to parser errors, so one avenue could involve improving the syntactic information in the data or introducing new extraction patterns designed to capture instances that might otherwise be missed due to an erroneous parse.

Another difficulty is related to the logical structure of certain shell nouns themselves. *Reason*, for instance, (cf. example (3)) appears to actually refer to two content phrases: one denoting a cause and the other an effect. *Möglichkeit* 'opportunity' seems likewise to have two parts: some thing that can happen and the circumstance that makes it possible. This semantic structure complicates both the annotation and resolution of these particular nouns.

The most likely explanation for the system's low recall and the most important avenue of future work relates to the lack of available training data for this task in general and for non-English languages in particular. If more data were available, a greater range of classification methods would become workable and significant performance improvements may be possible.

Alternatively, one could try to get more out of the existing data by allowing the classifiers to share certain information about the behavior of shell noun lemmas, in order to compensate for the fact that each lemma may only occur a handful of times in a particular pattern. Or one could use a lemma representation that better encodes semantic similarities, which might in turn help discover shell nouns that did not occur in the training data.

Most likely, improving the systems's performance will require both more annotated data and better use of the data that is available.

References

Bernd Bohnet, Joakim Nivre, Igor Boguslavsky, Richárd Farkas, Filip Ginter, and Jan Hajič. 2013. Joint morphological and syntactic analysis for richly inflected languages. *Transactions of the Association for Computational Linguistics*, 1:415–428.

Miriam Eckert and Michael Strube. 2000. Dialogue acts, synchronizing units, and anaphora resolution. *Journal of Semantics*, 17:51–89.

Birgit Hamp and Helmut Feldweg. 1997. GermaNet: A lexical-semantic net for German. In *Proceedings of the ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, Madrid, Spain.

Sujay Kumar Jauhar, Raul D. Guerra, Edgar Gonzàlez, and Marta Recasens. 2015. Resolving discourse-deictic pronouns: A two-stage approach to do *it*. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 299–308, Denver, CO, USA.

Varada Kolhatkar and Graeme Hirst. 2012. Resolving "this-issue" anaphora. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing*, pages 1255–1265, Jeju Island, Korea.

Varada Kolhatkar and Graeme Hirst. 2014. Resolving shell nouns. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 499–510, Doha, Qatar.

Varada Kolhatkar, Heike Zinsmeister, and Graeme Hirst. 2013. Interpreting anaphoric shell nouns using antecedents of cataphoric shell nouns as training data. In *Proceedings of the 2013 Conference on*

Empirical Methods in Natural Language Processing, pages 300–310, Seattle, Washington, USA. Association for Computational Linguistics.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4):885–916.

Ana Marasović, Leo Born, Juri Opitz, and Anette Frank. 2017. A mention-ranking model for abstract anaphora resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 221–232, Copenhagen, Denmark. Association for Computational Linguistics.

Christoph Müller. 2007. Resolving *it*, *this*, and *that* in unrestricted multi-party dialog. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 816–823, Prague, Czechia.

Christoph Müller. 2008. *Fully Automatic Resolution of* it, this *and* that *in Unrestricted Multi-Party Dialog*. Ph.D. thesis, Universität Tübingen.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. `scikit-learn`: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Massimo Poesio, Ron Artstein, Olga Uryupina, Kepa Rodriguez, Francesca Delogu, Antonella Bristot, and Janet Hitzeman. 2013. *The ARRAU Corpus of Anaphoric Information LDC2013T22*, volume Web Download. Linguistic Data Consortium, Philadelphia, PA, USA.

Hans-Jörg Schmid. 2000. *English Abstract Nouns as Conceptual Shells: From Corpus to Cognition*. de Gruyter, Berlin, Germany.

Fabian Simonjetz. 2015. Retrieving German shell nouns using dependency patterns. http://www.researchgate.net/publication/306020586_Retrieving_German_Shell_Nouns_Using_Dependency_Patterns.

Fabian Simonjetz and Adam Roussel. 2016. Crosslinguistic annotation of German and English shell noun complexes. In *Proceedings of the 13th Conference on Natural Language Processing (KONVENS)*, pages 265–278, Bochum, Germany.

Appendix

Feature	Examples
Shell noun	
Lemma	*Tatsache*, *Umstand*
Number	Sing./Pl.
Grammatical function	*subj*, *obja*
Whether parent precedes shell noun	Yes/No
Whether parent is subjunctive	Yes/No
Whether parent is clausal verb	Yes/No
Semantic field	*Attribut*, *Kommunikation*
Parent semantic field	*Gefühl*, *Perzeption*
Semantic fields of dep. adjectives	*Bewegung*, *Menge*
Whether dep. article is definite or indefinite	Yes/No
Dep. determiners	*dieser*, *kein*, *beiden*
Content phrase	
Dependent preposition lemmas	*zu*, *für*, *nach*
Dependent complementizers	*dass*, *ob*, *weil*
Grammatical function	*root*, *objc*
Length	No. of tokens
Gender	*Masc*, *Fem*, *Neut*
Semantic field	*Attribut*, *Kommunikation*
Embedding depth	No. of deps. to sentence root
If nominal, ending	*-ung*, *-heit*, *-en*
Contains question mark	Yes/No
Relation	
Distance between shell noun and content phrase	No. of tokens
Whether shell noun precedes content phrase	Yes/No
Whether grammatical functions match	Yes/No
Whether colon between shell noun and content phrase	Yes/No

Table 4: Complete list of features.