# Automatic Extraction of Typological Linguistic Features from Descriptive Grammars

Shafqat Mumtaz Virk[1(✉)], Lars Borin[1], Anju Saxena[2], and Harald Hammarström[2]

[1] Språkbanken, Department of Swedish, University of Gothenburg, Gothenburg, Sweden
virk.shafqat@gmail.com, lars.borin@svenska.gu.se
[2] Department of Linguistics and Philology, Uppsala University, Uppsala, Sweden
{anju.saxena,harald.hammarstrom}@lingfil.uu.se

**Abstract.** The present paper describes experiments on automatically extracting typological linguistic features of natural languages from traditional written descriptive grammars. The feature-extraction task has high potential value in typological, genealogical, historical, and other related areas of linguistics that make use of databases of structural features of languages. Until now, extraction of such features from grammars has been done manually, which is highly time and labor consuming and becomes prohibitive when extended to the thousands of languages for which linguistic descriptions are available. The system we describe here starts from semantically parsed text over which a set of rules are applied in order to extract feature values. We evaluate the system's performance on the manually curated Grambank database as the gold standard and report the first measures of precision and recall for this problem.

**Keywords:** Information extraction · Semantic parsing · Language typology · Typological database

## 1 Introduction

The area of linguistics which deals with comparison and classification of languages based on their structural features is known as linguistic typology. The objectives of this area are to find commonalities between, and to explore diversity across the world's languages, and to explain these in historical and/or universal terms. The typical flow of information in such investigations runs from primary linguistic data (collected in the field), to analyzed linguistic data (written down in descriptive grammars), to databases of features of interest (distilled into a language × feature matrix) which form input to the computational tools. The most widely known databases of linguistic structure include the *World Atlas of Language Structures* (WALS) (wals.info), the *Atlas of Pidgin and Creole Language Structures* (APiCS) (apics.org), the *South American Indigenous Language Structures* (SAILS) (sails.clld.org), AUTOTYP


K. Ekštein and V. Matoušek (Eds.): TSD 2017, LNAI 10415, pp. 111–119, 2017.
DOI: 10.1007/978-3-319-64206-2_13

(github.com/autotyp/autotyp-data), and the *Phonetics Information Base and Lexicon* (PHOIBLE) (phoible.org). A fuller listing of available linguistic databases is provided at languagegoldmine.com/.

To the best of our knowledge, all the linguistic databases published so far have been manually constructed, where human experts have turned information from field data or analyzed data into datapoints in the database. The use of human expertise guarantees a certain level of quality and robustness, but is highly labor intensive and consequently costly. There are some 6,500 languages in the world, out of which descriptive grammars – ranging from brief grammar sketches to multi-volume reference grammars – are available for over 4,000 (see glottolog.org). Manually extracting information about 200–300 features from each of them is a very ambitious – and in practice unrealistic – undertaking.

Significant amounts of analyzed language data (grammatical descriptions in discursive textual form) are increasingly being made available in digital form, and the field of natural language processing (NLP) offers tools that potentially can aid us in extracting information about linguistic features from such textual sources, at least for sources in English and some other languages. To take advantage of these advancements, and to help the linguistic community in populating the linguistic feature databases, we report on a pioneer system for automatically extracting information about selected linguistic features from descriptive grammars. The system is based on semantic parsing and a set of rules to extract and formulate the required information. We test our system on Grambank data taking it as a gold standard, and report first measures of precision and recall in this direction.

## 2 Data

The work presented in this paper has been conducted as part of a larger long-term research project where work is ongoing on compiling a comprehensive database of linguistic features from the *Linguistic Survey of India* (LSI, a massive text data source) [3], in order to investigate areal-linguistic features of the languages of South Asia. In part, this database is to be used for evaluating the NLP tools for automatic feature extraction developed in the project. Since the tools themselves are intended to be applicable to more than one data source, we are in the meantime drawing on another source of gold standard data, using linguistic features also relevant for the South Asian languages. In the experiments reported here, we have used a subset of the Grambank data to test and evaluate the system. Grambank is the name of a database of structural (typological) features being developed at the Max Planck Institute for the Science of Human History at Jena. It surveys 195 structural features of language drawing on and extending the set used by [6]. The scope of Grambank is worldwide and it currently contains data for over 700 languages. The main advantage of using Grambank is that the descriptive grammars consulted in compiling it are both known and available to us in digital form, a set of approximately 6,000 digital descriptive grammars.

**Table 1.** Semantic parse

| Predicate | Semantic arguments |
|---|---|
| follow | ARG1: The_adjectives, ARG2:the_noun_they_qualify |
| qualify | ARG1: the_noun_they |

For our experiments, we preprocessed the available set of digital grammars as follows. First, we removed all those documents whose language of description was not English (since the semantic parser that we are using in this study is for English, we have restricted our experiments to English-language documents only). From the remaining document set, we grouped all those documents which were the source of the value for a particular feature into one subset. This gave us five subsets of documents, one for each of our five target features listed in Sect. 3. Cases where there was more than one source document for a given feature in a particular language were removed from consideration as they were not corresponding to the LSI use case (with only one description per language). Further, if a document was a source of the same feature for multiple languages, it was filtered out, since for those cases an extra step is required to map the feature value to the language, which we left for future work.

## 3 Automatic Feature Extraction and Formulation

In this study, we have targeted the following features: (1) Apos: What is the order of adnominal property word and noun?[1] (2) NLpos: What is the order of numeral and noun in the NP? (3) AagrNum: Can an adnominal property word agree with the noun in number? (4) AagrGen: Can an adnominal property word agree with the noun in gender? (5) DefArticle: Are there definite or specific articles?

Though the focus in this article is only on these features, it is worth mentioning that the proposed methodology can also be used easily to extract information about other features too. The procedure of automatically extracting feature values is as follows: As a first step, a given reference grammar was sentence segmented using the Natural Language Tool Kit (NLTK).[2] Each of the sentences was then parsed using a Propbank based semantic parser [1]. From each parsed sentence a list of predicates and their semantic arguments were extracted. The predicates and their arguments were then further analyzed to examine if a particular predicate and its semantic arguments contain the information we are interested in (i.e. information about a particular linguistic feature). Additional analysis steps included: (1) checking for particular predicates for particular features; (2) inspecting the semantic arguments' structure and contents; and

[1] An *adnominal property word* corresponds to an adjective or participle in English and many other languages.

[2] http://www.nltk.org/.

**Table 2.** Linked predicate set for features

| Feature | Linked-predicate set |
|---|---|
| Apos, NLpos | (i) follow (ii) precede (iii) come (iv) place |
| AagrNum, AagrGen | (i) agree (ii) inflect (iii) change |
| DefArticle | (i) be (ii) use (iii) lack |

(3) formulating the feature values. Let's take an example to illustrate the whole procedure. Suppose we are interested in extracting information about adjective–noun order for a particular LSI language, e.g., Siyin.[3] In the descriptive grammar used for this language, the information about adjective–noun order has been conveyed through the sentence *The adjectives follow the noun they qualify*. Parsing this sentence using the semantic parser will return a set of predicates and their semantic arguments which are listed in Table 1. The predicate 'follow' is one of those predicates that were identified, as a separate process, to be linked to the adjective–noun order feature. Using a development data set, we identified a set of predicates linked to each of target features. This simply involved finding sentences in the descriptive grammars which were used to provide information about a particular feature, and then analyzing them to find the associated list of predicates. Table 2 contains the list of predicates that were identified for each of the target features.

The next step is to examine the semantic arguments of the predicate 'follow', and formulate the feature value. As per Propbank, for the predicate 'follow', ARG1 represents the thing following, while ARG2 represents the thing followed. In the analysis shown in Table 1, the string *The adjectives*, is ARG1 (i.e. the thing following), while the string *the nouns they qualify* is ARG2 (i.e. the thing followed). The substrings representing ARG1, and ARG2 can be further analyzed to formulate and return the feature value '2-N-ANM' (the fact that adjectives follow the nouns). Had ARG1 contained *noun(s)*, ARG2 contained *adjective(s)* with predicate being 'follow', or ARG1 contained *adjective(s)*, ARG2 contained *noun(s)*, and the predicate being 'precede', '1-ANM-N' (the fact that adjectives precede nouns) would have been returned as the feature value. We have used simple if-then-else condition based rules to examine predicates and their semantic argument strings for the purpose of extracting and formulating the feature values. In the future, we have plans to experiment with more advanced techniques, such as active learning, for the feature extraction and formulation from the semantic parses.

A simplified version of the algorithm that was used to extract the adjective–noun order feature values is given in Algorithm 1. As can be noted, the algorithm simply loops over the set of predicates (line 4), and for each predicate it collects the numbered and modifier arguments (lines 5–6). If the predicate is one of the linked predicate for the target feature (line 7), it loops through

[3] A Tibeto-Burman language of Burma with about 10,000 speakers.

the list of (argument_label, argument_string) pairs (line 12). At each iteration, the argument_label and argument_string[4] are examined and appropriate boolean variables are adjusted (lines 13–21). Once we have examined all semantic arguments of a predicate, and have adjusted the variables, different combinations of these variables are tested to adjust different order-specifying variables (lines 23–33). For example, if 'adjective' is the agent of the predicate 'follow', and 'noun' is the patient, it means 'adjective follow noun'. Similarly, if 'noun' is the agent, and 'adjective' is the patient, it means 'adjective precede noun'. Also note, how the contents of modifier arguments are tested. For example, if we have a sentence *Adjectives usually follow nouns*, it means adjective may follow or may precede the noun, and the system should be able to return 'both' as a feature value. The algorithm takes care of such cases with an extra condition analyzing the contents of modifier arguments (lines 25–27 and 30–32). Finally, the algorithm formulates the feature values in the required format and returns them back (lines 36–42). For clarity of exposition, the algorithm shows the handling of the predicate 'follow' only, but it is easy to think of a similar sort of handling for other linked predicates (given in Table 2) for each of the target features discussed in this study.

The sentence containing the description of a particular feature may have anaphoric expressions referring to an antecedent or subsequent expression. For example, the sentence *They follow nouns* may appear instead of *Adjectives follow nouns* in the description, with the antecedent expression *adjectives* appearing somewhere else. To extract feature values from such sentences properly, such anaphoric relations need to be resolved. There exist many anaphora resolution systems [2,5], and a classical solution will involve using a state-of-the art system for this purpose. At the current stage of our experiments, however, we have chosen to employ a simple rule-based strategy to resolve such co-reference relations and extract feature values/descriptions. The main idea is to investigate the context with a particular window size to resolve such co-references (if any). For example, if a semantic argument is a pronoun (e.g. *they*, *it*, or *them*), we just investigate the semantic arguments of one or more preceding or following sentences, and if those arguments contain the potentially linked entity (e.g. *nouns* or *adjectives*, etc.), we just assume that they are related to each other. This procedure can easily be incorporated in Algorithm 1 with an extra if-else condition, but for simplicity, we have excluded it here. It is worth mentioning here that the rule-based anaphora resolution solution was chosen not only for its simplicity, but we also observed in experiments that in many cases this simple strategy was actually able to relate the arguments, whereas the Stanford anaphora resolution system [5] failed to do that.

## 4 Evaluation

The evaluation results are given in Table 3. As can be seen, the system has varying precision and recall for different features, which highlights the difficulty/ease

[4] The argument string is split into a set of words using NLTK's word tokenizer.

**Algorithm 1.** Extract Adjective Noun Order

```
1: procedure ExtractAdjectiveNounOrder(parses)
2:     AdjectiveFollowNoun ← False
3:     AdjectivePrecedeNoun ← False
4:     for <every predicate in parses> do
5:         NumberedArgs ← NumberedArgumentsOfPredicate
6:         ModifierArgs ← ModifierArgumentsOfPredicate
7:         if predicate = follow then
8:             AdjectiveAgentFollow ← False
9:             NounPatientFollow ← False
10:            NounAgentFollow ← False
11:            AdjectivePatientFollow ← False
12:            for <every (ArgStr,ArgLabel) of NumberedArgs> do
13:                if ArgLabel = A1 ∧ adjective ∈ ArgStr then
14:                    AdjectiveAgentFollow ← True
15:                else if (ArgLabel = A2 ∧ noun ∈ ArgStr) then
16:                    NounPatientFollow ← True
17:                else if ArgLabel = A1 ∧ noun ∈ ArgStr) then
18:                    NounAgentFollow ← True
19:                else if ArgLabel = A2 ∧ adjective ∈ ArgStr then
20:                    AdjectivePatientFollow ← True
21:                end if
22:            end for
23:            if AdjectiveAgentFollow ∧ NounPatientFollow then
24:                AdjectiveFollowNoun ← True
25:                if usually ∈ ModifierArgs ∨ sometimes ∈ ModifierArgs then
26:                    AdjectivePrecedeNoun ← True
27:                end if
28:            else if NounAgentFollow ∧ AdjectivePatientFollow then
29:                AdjectivePrecedeNoun ← True
30:                if usually ∈ ModifierArgs ∨ sometimes ∈ ModifierArgs then
31:                    AdjectiveFollowNoun ← True
32:                end if
33:            end if
34:        end if
35:    end for
36:    if AdjectiveFollowNoun = True ∧ AdjectivePrecedeNoun = True then
37:        return '3 − both'
38:    else if AdjectiveFollowNoun = True ∧ AdjectivePrecedeNoun = False then
39:        return '2 − N − ANM'
40:    else if AdjectiveFollowNoun = False ∧ AdjectivePrecedeNoun = True then
41:        return '1 − ANM − N'
42:    end if
43: end procedure
```

of automatically extracting the corresponding feature values using the described method. The 'NLPos' feature has the best precision while the 'Apos' feature has the best recall value. The next section contains a detailed error analysis, discussion of possible reasons for the low recall, and suggestions for how to improve both precision and recall.

As mentioned previously, there does not exist any work related to automatic linguistic feature extraction, which means we do not have any other system to compare the proposed system's performance with. Instead, we evaluate the system performance against a baseline calculated for each feature on the basis of the most frequent feature value in the entire Grambank database. As can be noted, for four out of the five features, the proposed system was able to easily beat the baseline precision values, the exception being the feature 'AagrNum'.

**Table 3.** Evaluation results

| Feature | Precision | Recall | F-score | Baseline precision |
|---|---|---|---|---|
| Apos | 0.76 | 0.40 | 0.52 | 0.41 |
| NLpos | 0.85 | 0.30 | 0.44 | 0.75 |
| AagrNum | 0.69 | 0.21 | 0.32 | 0.77 |
| AagrGen | 0.64 | 0.14 | 0.23 | 0.27 |
| DefArticle | 0.84 | 0.27 | 0.41 | 0.27 |

## 5 Error Analysis and Discussion

We did a detailed error analysis and found that there are three major sources of errors: (1) debatable gold feature values; (2) pre-processing and semantic parsing errors; (3) a grammar of one language including information about another language.

In many of the erroneous cases, the output produced by the proposed system actually seems reasonable. For example, for the feature Apos, the system determined the value to be '3-both' based on the modifier argument *usually* in the sentence *'An attributive adjective usually precedes (comes before) the noun that it modifies'* from the grammar of the Pulaar[5] language. In the gold data, the value is listed as '2-N-ANM'. Another similar case is in the grammar of the language Pech,[6] where it is stated that *Both qualitative and quantitative adjectives follow the nouns they modify; however, demonstrative adjectives precede nouns.* Based on this description the system determined the value to be '3-both', but in the gold standard, the human experts have determined the feature value to be '2-N-ANM'. Similar cases were found for other features as well. Such cases need to be investigated further. There are two possibilities: (1) either the feature value in the gold standard is debatable (or wrong) – there are studies suggesting that the accuracy of manually curated typological databases is not always 100% and it may vary from 80 to 90% [4] – or (2) the information has been formulated in a different or an indirect way in the grammar, which the system was unable to extract. This needs to be further investigated. Whatever the reason may be, one can use the system proposed in this study to do a kind of validation of the gold data, which can be considered as an additional use-case of the system.

Sentence segmentation errors may propagate all the way from semantic parses to the output produced by the proposed system. For example, consider the following sequence of sentences which was (erroneously) not segmented and passed to the semantic parser as a single sentence:

*An indefinite noun phrase negated by md may also be preceded by cdd "still, etc." 5 or gad, as in : ma cad halib yirjac darrih md gad yahudi nasah muslim "milk will not return to the breast" Z.12/51 "no Jew has advised a Muslim" Indefinite adjectives are likewise negated by ma in predicand position (cf.*

[5] An Atlantic-Congo language spoken in Africa.
[6] A Chibchan language spoken in Central America.

In the resulting semantic parse, the predicate 'precede' was identified having arguments (1) ARG0, which contained the string *adjectives*, and (2) ARG1, which contained the string *nouns*. This caused the feature formulation algorithm to return the wrong feature value '1-ANM-N' (i.e. adjective precede nouns).

It also happens that a grammar contains information about other languages than the main language being described, which may become a source of errors. For example, consider the following sentence from the reference material of a language called Tu: *Numerals in Baonan may either precede or follow a head noun.* The sentence provides information about the language Baonan, but the proposed system will extract and mark it for the language Tu. A solution to this issue could be to identify the language name Baonan first, and then link the extracted information to this language – a task that we leave for future work.

While the precision is relatively good, recall seems too low. We have identified two probable reasons for this: (1) At times, the information about a particular feature is presented indirectly in a descriptive grammar, which is easy for a human to extract but difficult for the automatic system. For example, the grammar may provide language examples with noun phrases containing adnominal property words, which could be used by a human to find about the adjective–noun order in a language, even in the absence of an explicit statement. Since the current system does not extract such indirect information, this may contribute towards a low recall. (2) Due to the small development data-set, it is very likely that we could not find out all the ways and linked predicates (Table 2) which might have been used to encode information about particular features. This, too, can be a factor contributing towards low recall. With the availability of more development data, we believe the system's recall can be considerably improved.

## 6 Conclusions and Future Work

Given that the manual compilation and curation of typological databases are very labor and time consuming[7] enterprises, any assistance for doing it (semi)automatically is a welcome addition. In a semi-automatic setting, such a system can be used to get pointers to the relevant text segments which may contain the required information. Once these have been pinpointed, it becomes a lot easier to manually extract the feature values from language descriptions which may be as long as a multi-volume book.

In this study, we have reported our experiments related to automatic extraction of linguistic features, and first measures of precision and recall. Since we don't have any other systems to compare our results with, we have shown improvements over an expected value based baseline system.

The most urgent next step is to improve the recall. As mentioned above, the recall can be improved given that we have more development data to find other ways in which information about features could have been encoded. As this depends on development and availability of more data, we have another

[7] In a separate study we found that the average cost for a salaried student assistant to extract one datapoint from a descriptive grammar is 1.53 EUR.

possibility in mind, and that is to ask the assistants, who are working in our LSI project, as well as in the Grambank or other typological database projects to record the sentence(s) which they think contain the relevant information. Later, the sentence structure and contents can be used to enhance the system's ability to automatically extract more information and hence to improve the system's recall. As a long term goal, we would like to extend the experiments to other linguistic features (ideally to all of Grambank's 195 linguistic features).

## References

1. Björkelund, A., Hafdell, L., Nugues, P.: Multilingual semantic role labeling. In: Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task, CoNLL 2009, pp. 43–48. Association for Computational Linguistics, Stroudsburg (2009)
2. Broscheit, S., Poesio, M., Ponzetto, S.P., Rodriguez, K.J., Romano, L., Uryupina, O., Versley, Y., Zanoli, R., Kessler, F.B.: Bart: a multilingual anaphora resolution system. In: Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval 2010, pp. 104–107 (2010)
3. Grierson, G.A.: A Linguistic Survey of India, vol. I–XI. Government of India, Central Publication Branch, Calcutta (1903–1927)
4. Polyakov, V.N., Solovyev, V.D., Wichmann, S., Belyaev, O.: Using wals and jazyki mira. Linguist. Typology **13**, 137–167 (2009)
5. Raghunathan, K., Lee, H., Rangarajan, S., Chambers, N., Surdeanu, M., Jurafsky, D., Manning, C.: A multi-pass sieve for coreference resolution. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP 2010, pp. 492–501. Association for Computational Linguistics, Stroudsburg (2010)
6. Reesink, G., Singer, R., Dunn, M.: Explaining the linguistic diversity of sahul using population models. PLoS Biol. **7**(11), 1–9 (2009)