# Manual and Automatic Tagging of Indo-Aryan Languages

**Rafał Jaworski, Krzysztof Jassem, Krzysztof Stroński**

Adam Mickiewicz University
Wieniawskiego 1, 61-712 Poznań, Poland
{rjawor,jassem,stroniu}@amu.edu.pl

**Abstract**

An annotation tool called IA Tagger, used for semi-automatic tagging of New Indo-Aryan texts, is presented. One of the features of the system is the generation of statistical data on occurrences of words and phrases in various contexts, which helps perform historical linguistic analysis at the levels of morphosyntax, semantics and pragmatics. The paper also reports on two experiments carried out on data annotated with the use of IA Tagger, involving the training of multi-class and binary POS-classifiers.

## 1. Introduction

The aim of the present paper[1] is twofold. Firstly, it attempts to give a brief overview of a tool designed for semi-automatic annotation of early New Indo-Aryan (NIA) texts. Secondly, it focuses on several aspects of automatic POS-tagging of early NIA.

There has been a considerable amount of corpus-based research into Old and New Indo-Aryan, which has already contributed much to our knowledge and understanding of the history of one of the main branches of Indo-European. However, there is still a need for research into early NIA. The present paper is a modest contribution to the corpora collation and preliminary analysis of early NIA morphosyntax from various perspectives. We decided to focus on selected early NIA tongues such as Rajasthani, Awadhi, Braj, Dakkhini and Pahari, and we present here the preliminary results of research into the Rajasthani language, based on short prose texts from the 15th to 18th centuries supplemented by early Awadhi poetry[2].

## 2. The IA Tagger tool

### 2.1. Tool overview

IA Tagger is a tool for text annotation specifically for Indo-Aryan languages. The key functionality of the tool is multi-level annotation of words and sentences of early NIA texts (see Section 2.2.). IA Tagger provides several features that improve the efficiency of use. For most annotation levels the system displays a context-sensitive list of prompts of available annotation tags. For a word under annotation the system displays a "prompt cloud", which consists of a set of tag suggestions (see Section 2.3.).

IA Tagger minimizes the cost of usage errors or system failure. Each annotation decision is saved automatically in a periodically backed-up database. There is no save button.

The wide variety of configuration settings ensures the flexibility of the tagger, allowing it to be used in various scenarios (see Section 2.4.).

On request IA Tagger generates statistics concerning occurrences of specific classes of words and word collocations – in a specified document or collection of documents (see Section 2.5.).

The system is intended for open access. It is accessible using any popular Internet browser at *http://rjawor.vm.wmi.amu.edu.pl/tagging*. Access credentials can be obtained on request from *rafal.jaworski@amu.edu.pl*.

### 2.2. Multi-level tagging

Upon upload to the system, a document is automatically split into sentences (see Figure 1).

The user can easily override the automatic sentence split (using "scissors" or "glue"; Figure 1). The document is annotated in sentence-by-sentence mode.

Each sentence is automatically split into words. The user may override the word split, e.g. in order to divide a word into a stem and a suffix. Words are annotated at six levels: Lexeme (where the closest English lexical equivalent is given), Grammar (annotated by means of Leipzig Glossing Rules), POS (Parts of Speech), Syntax (exploring the basic Dixonian (Dixon, 1994) scheme based on the three primitive terms: A, S and O, where A stands for the subject of a transitive sentence, S for the subject of an intransitive sentence and O for the object of a transitive sentence), Semantics (where we distinguish six basic thematic roles: Agent, Patient, Experiencer, Recipient, Stimulus and Theme, based on the RRG approach, e.g. (Van Valin, 2005)), and Pragmatics (distinguishing Topics).

Figure 2. represents an annotated sentence in Rajasthani.

### 2.3. Automatically generated suggestions

In order to improve tagging efficiency, the system suggests hints whenever possible, i.e. when a word has already been tagged or when the tagging could be deduced automatically. Tag suggestions appear in a "cloud" above the word (Figure 3).

Figure 3 shows tag suggestions for the word 'nagarli' (the pipe indicates that the word 'nagari' has been split into a stem and a suffix). The first two lines come from previous annotations, whereas the third line is the set of suggestions deduced automatically. The user can accept the set of suggestions by clicking the 'check' symbol in the left-most column. The annotation shown in Figure 2 was obtained by applying the third-line set of tags.

[1]This paper is a part of a research project funded by Polish National Centre for Science Grant 2013/10/M/HS2/00553.

[2]Optical recognition of Rajasthani texts was supported by a Hindi OCR program (Hellwig, 2015)

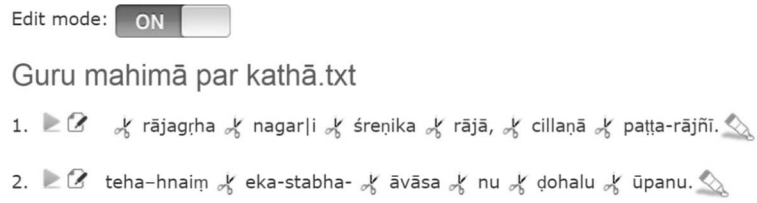


Figure 1: Sentence split.

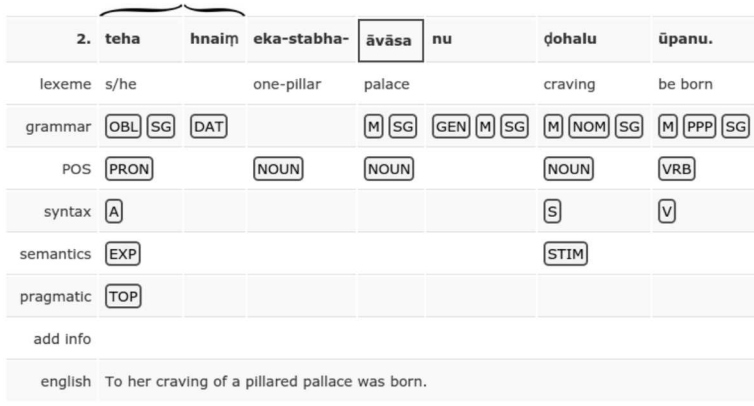


Figure 2: Annotated sentence.


Figure 3: Automatically generated suggestions.

### 2.4. Configuration

IA Tagger may be configured to serve a variety of annotation tasks. The "configuration" option allows one to manage the languages of tagged documents as well as to configure annotation levels. Annotation levels may be freely ordered, added, deleted or edited. Editing of an annotation level consists in defining admissible values of respective tags.

### 2.5. Flexible statistics generator

On request, IA Tagger generates statistics concerning occurrences of specific classes of words and word collocations – in a specified document or collection of documents. This facilitates linguistic analysis at the levels of syntax, semantics and pragmatics. Initial analysis assumes a survey of alignment features, i.e. main argument marking (A and O). This kind of research has already been carried out by several authors for finite verbs (e.g. (Khokhlova, 2001)), but IA Tagger makes it possible to generate statistics for texts belonging to various historical stages of NIA, and it has a much larger scope since it encompasses both finite and non-finite verbs such as converbs[3], infinitives and participles. Preliminary research on a Rajasthani annotated corpus shows a preponderance of unmarked O forms over marked ones. Out of 201 converbal chain constructions only 18 show marked O forms. Up to the 18th century there are only examples of animate and definite Os, or possibly human and indefinite, and from the 18th century onwards inanimate Os start appearing (see ex. (1)).

(1) Old Rajasthani, 18th c. (Bhanavat and Kamal, 1997-1998, 72)

| | | |
|---|---|---|
| tiṇa | sahanāṇa-ṇūṃ | dekha |
| that.OBL.SG | sign-ACC | see.CVB |

'having seen that sign'

This conforms nicely with Khokhlova's findings for finites and with more general tendencies operating along definiteness and animacy hierarchies (cf. for example (Aissen, 2003)).

The next steps of the analysis will consist in multifaceted analysis of IA non-finites (focusing on converbs and infinitives) drawing from two frameworks: RRG (Van Valin and LaPolla, 1997), (Van Valin, 1993), (Van Valin, 2005) and Multivariate Analysis (Bickel, 2010) where apart from morphosyntactic properties,

[3]We accept here Haspelmath's (Haspelmath, 1995, 3) definition of the converb: "a nonfinite verb form whose main function is to mark adverbial subordination".

semantic and pragmatic properties of converbs will be investigated. Semantic analysis involves checking the control properties of converbs and their stability or fluctuation in the history of NIA, whereas pragmatic analysis is based on investigation of the scopal properties of selected operators in converbal chain constructions. Preliminary results of the latter analysis were presented in (Stroński and Tokaj, 2015). It has been observed that Illocutionary Force Operator can have conjunct or local scope, and this property is quite stable throughout the centuries, whereas Tense Operator seems to have conjunct scope in those converbal chains which have the main verb in the past tense and almost exclusively local scope in those chains which have the main verb in the present tense (cf. ex. (2) and (3)). This somehow implicitly presumes the perfectivity of the IA converb, which in turn appears to be a historically important finding.

(2) Old Rajasthani, 15th c. (Bhanavat and Kamal, 1997-1998, 15)

| | | | |
|---|---|---|---|
| āmbā | leī̄ | ḍohalu | pūriu |
| mango.NOM.M.PL | take.CVB | craving.M.SG | fill.PPP.M.SG |

'having taken mangos, fulfilled the craving'
'took mangos and fulfilled the craving'

(3) Old Rajasthani, 18th c. (Bhanavat and Kamal, 1997-1998, 61)

| | | | |
|---|---|---|---|
| phūladhārā | vica | uḍi | paṛaṃ |
| stream of flowers | middle | fly.CVB | fall.1PL.PRS |

'having flown in the middle of the stream of flowers, we will fall'

## 3. Automatic POS-tagging

### 3.1. Similar experiments

Experiments with automatic POS-tagging of less-resourced languages have already been conducted in recent years. This subsection briefly describes the techniques used and the outcome of two projects: an automatic tagger for Urdu, developed by (Hardie, 2005), and Sanskrittagger by (Hellwig, 2008).

#### 3.1.1. Urdu tagger

The tagger for Urdu was developed by Andrew Hardie in 2005. The main difficulty in tagging Urdu texts identified by the author was word sense disambiguation. Two techniques were implemented in order to resolve this problem. One was based on hand-crafted rules prepared by a linguist, while the other relied on statistical analysis of manually annotated Urdu texts. The author reports the low effectiveness of the latter method, attributing it to the relatively small quantity of training data. Hence the author decided to use the tagger based on hand-crafted rules. It must be pointed out, however, that the statistical model used was HMM (Hidden Markov Models), which was considered state-of-the-art in the early 2000s, but was replaced in the following years by several other methods, such as Conditional Random Fields or Maximum Entropy.

The resulting rule-based tagger used a tagset of approximately 80 tags and achieved an accuracy of 88–90%. The author admitted that these results were lower than those of taggers for well-resourced languages, such as English. Such taggers score at least 95% accuracy. This, however, should not be considered the main flaw of this system. A more important drawback of the approach presented by Hardie is the heavy reliance on manually designed rules, which account for most of the positive results of the system. These rules were specially designed to work with Urdu, and even more specifically – with the Urdu texts that were at the author's disposal. In a different scenario the same rules may prove to be inapplicable, thus impairing the performance of the system significantly.

#### 3.1.2. Sanskrittagger

Sanskrittagger, described in (Hellwig, 2008), is an automatic tokenizer and tagger for Sanskrit. Like Hardie's Urdu tagger, it uses HMM to perform the tagging. Interestingly, the same model is also applied to the task of tokenization, which is a non-standard solution.

The system uses a tagset of 136 tags. Unfortunately, accuracy figures are not known, as the evaluation of the system was performed on only five short passages of text. However, it is revealed that the system is purely statistical.

Among suggested methods of improvement, one seems particularly interesting – integrating tokenization and POS-tagging into one mechanism. The author argues that this might be a good approach for Sanskrit, even though it is not commonly used for other languages.

### 3.2. Training the automated tagger

The IA Tagger system has been used by a team of linguists for several months. The work has resulted in a manually annotated corpus of the early Rajasthani language supplemented by early Awadhi. The corpus so far contains 1284 sentences with 13 022 words. Even though the size of the corpus is too small for statistical data analysis, experiments were run to determine whether it is possible to create a usable POS tagger for early NIA.

Firstly, two separate POS tagging systems were developed. One of them uses a set of 22 tags to annotate the text. The tags are hierarchical, e.g. there is a NOUN tag and its child – NOUN-SINGULAR.

The other tagger is a detector of specific verb forms – converbs.

#### 3.2.1. Multi-class POS tagging

The task of annotation with 22 tags was seen as a multi-class classification problem. In order to implement such a tagger, a well-known Maximum Entropy tagging mechanism was used. This idea was first proposed by (Ratnaparkhi, 1996) and later used to implement the Stanford Part-Of-Speech Tagger (see (Toutanova and Manning, 2000) and (Toutanova et al., 2003)). The automatic tagger for early NIA is based on the Stanford software.

The main difficulty in training automatic taggers using the Maximum Entropy principle is the identification of the feature set. Possible features may include: suffix($n$) of the word (i.e. last $n$ letters), length of the word, whether the word starts with a capital letter (boolean feature) and many others. It is crucial, however, that all these features should

| Metric | Correct tags # | Accuracy |
|---|---|---|
| exact | 6210 | 57.9% |
| partial | 6874 | 64.1% |

Table 2: Overall results of the multi-class tagger

be computable on unannotated text. Thus, features like "is located between a noun and a verb" are not acceptable.

The described automatic tagger for early NIA texts uses the following set of features: *Suffix(6)*, *Previous word (i.e. the literal text form of the previous word)*, *Next word* and *Distributional similarity class*.

Distributional similarity (often abbreviated $distsim$) is a method for categorizing words in a large corpus based on their contexts. Each word falls into a category with other words that appeared in similar contexts. The id of such a category can be used as a word feature.

In order to compute distributional similarity classes, an unannotated modern Rajasthani corpus of 81 843 words was used. It was processed with the help of word2vec software, described in (Mikolov et al., 2013). The words were categorized into 209 classes, each containing between 1 and 66 words. For example, one of the classes contained the following words: *te* 'this', *teha* 's/he', *bi* 'two', *bewai* 'both', which are all pronouns.

#### 3.2.2. Converb detector

The second approach involved the training of a separate Maximum Entropy tagger, focused solely on identifying words of special interest – converbs. This is a case of binary classification. The implementation of the converb detector is based on the Python NLTK library, described in (Loper and Bird, 2002), which is capable of using an optimization technique called MEGAM (Daumé III, 2004). This makes it possible to create a robust binary classifier.

The converb detector was trained on the same data as the multi-class tagger described in Section 3.2.1. The features used by this detector are presented in Table 1. Note that the features $cvbEnding$ and $firstOrLast$ use linguistic knowledge about converbs. Firstly, Rajasthani converbs typically terminate in /i/ and /a/, although from the earliest texts onwards other suffixes are also attested. Secondly, converbs would never appear as the first or last word in the sentence. This approach recalls the hand-crafted rules as seen in (Hardie, 2005). However, the features are never strict. The decision on whether or not to use a specific feature is made by the statistical model.

### 3.3. Experiment results

This section presents the results of the experiment conducted using both of the automatic POS taggers. In both cases the tagged corpus (13 022 words) was used to perform 10-fold cross-validation.

Table 2 presents results for the multi-class tagger. It assigned tags to 10 730 out of 13 022 words (82.4%), leaving the remaining words untagged. Exact tag matching counts a tag as correct only if it matches exactly the tag in the golden standard. Partial tag matching allows, for example, the tagging of a NOUN-SINGULAR with the tag NOUN.

| Word form | Precision | Recall | F-measure |
|---|---|---|---|
| Verb | 0.61 | 0.70 | 0.65 |
| Noun | 0.41 | 0.52 | 0.46 |
| Past participle | 0.70 | 0.60 | 0.64 |
| Converb | 0.33 | 0.07 | 0.11 |

Table 3: Detailed performance of the multi-class tagger

| Metric | Value |
|---|---|
| Precision | 0.83 |
| Recall | 0.39 |
| F-score | 0.53 |

Table 4: Converb detector scores

Selected POS classes were investigated more thoroughly. Table 3 presents precision, recall and F-measure scores (as proposed in (Makhoul et al., 1999)) for identifying these forms. All results assume the partial tag matching metric.

The overall accuracy of the tagger (64% – Table 2) is not satisfactory, which indicates the need for using larger data sets in the learning process. Table 3 shows that converbs, which are of particular interest in our research, are recognized with lower accuracy than other classes. This inspired further study using the specialized converb detector.

The detector was expected to attain higher precision and recall scores in finding converbs than the multi-class tagger. Its scores are presented in Table 4. These indeed show a considerable improvement over the multi-class tagger (see Table 3). This justifies the decision to implement a separate detector solely for word forms of particular interest.

## 4. Conclusions and future work

The paper demonstrates how IA Tagger – a semi-automatic annotating tool – can help perform multi-level historical linguistic analyses pertaining to morphosyntax, semantics and pragmatics. A flexible statistics generator facilitates distributional analysis of various converbal forms and analysis of main argument marking with finite and non-finite verbs. At the semantic level it can also support analysis of the control properties of converbs, and at the pragmatic level it clearly helps establish the scope of main clause level operators.

In the future, the IA Tagger can further support research on other non-finite verb forms such as infinitives and participles, and what is more, it can easily identify the main grammaticalization paths with respect to light verbs.

This paper reports also on preliminary research on automatic POS-tagging for the early Rajasthani language. The research consisted in two experiments carried out on a small set (13 022 words) of annotated data. The first study had the aim of creating a multi-class POS classifier trained on the available data. The second investigated the accuracy of a binary classifier devoted to a class that was recognized poorly in the first experiment. The experiments applied

| Feature name | Parameters | Description |
|---|---|---|
| word | none | literal text of the word |
| wordContext | $n$ | $n$ words to the left and $n$ words to the right of the word |
| suffix | $n$ | $n$ last characters of the word |
| class | none | distributional similarity class |
| classContext | $n$ | as in wordContext, only on $distsim$ classes |
| cvbEnding | none | whether or not the word ends in a typical converb ending |
| firstOrLast | none | whether or not the word is first or last in the sentence |

Table 1: Features used by converb detector

standard machine learning techniques, including the recently investigated idea of distributional similarity classes. The evaluation results proved that applying purely statistical methods to a small corpus of annotated data does not yield practically applicable results for multi-class recognition. The binary classifier, however, achieved a high value for the precision measure. We conclude that applying state-of-the-art machine learning techniques to languages that lack large annotated corpora is currently useful only for binary classification.

## 5. References


Aissen, Judith, 2003. Differential object marking: iconicity vs. economy. *Natural Language and Linguistic Theory*, 21:435 – 483.

Bhanavat, Narendra and Lakshmi Kamal, 1997-1998. *Rajasthani gadya: vikas aur prakash*. Shriram Mehra and Company.

Bickel, Balthasar, 2010. Capturing particulars and universals in clause linkage: a multivariate analysis. In Isabelle Bril (ed.), *Clause Linking and Clause Hierarchy : Syntax and Pragmatics*, number 121 in Studies in Language Companion Series. Amsterdam: John Benjamins, pages 51 – 102.

Daumé III, Hal, 2004. Notes on CG and LM-BFGS optimization of logistic regression.

Dixon, Robert M.W., 1994. *Ergativity*. Cambridge Studies in Linguistics. Cambridge University Press.

Hardie, Andrew, 2005. Automated part-of-speech analysis of Urdu: conceptual and technical issues. *Contemporary issues in Nepalese linguistics*:48 – 72.

Haspelmath, Martin, 1995. The converb as a cross-linguistically valid category. In Martin Haspelmath and Ekkehard König (eds.), *Converbs in Cross-Linguistic Perspective: Structure and Meaning of Adverbial Verb Forms – Adverbial Participles, Gerunds*, number 13 in Empirical approaches to language typology. Berlin: Mouton de Gruyter, pages 1 – 55.

Hellwig, Oliver, 2008. Sanskrittagger: A stochastic lexical and POS tagger for Sanskrit. In Gérard P. Huet, Amba P. Kulkarni, and Peter M. Scharf (eds.), *Sanskrit Computational Linguistics*, volume 5402 of *Lecture Notes in Computer Science*. Springer.

Hellwig, Oliver, 2015. ind.senz – OCR software for Hindi, Marathi, Tamil, and Sanskrit. http://www.indsenz.com.

Khokhlova, Liudmila, 2001. Ergativity attrition in the history of Western New Indo-Aryan languages (Punjabi, Gujarati and Rajastahani). *The Yearbook of South Asian Languages and Linguistics*:159 – 184.

Loper, Edward and Steven Bird, 2002. NLTK: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02. Stroudsburg, PA, USA: Association for Computational Linguistics.

Makhoul, John, Francis Kubala, Richard Schwartz, and Ralph Weischedel, 1999. Performance measures for information extraction. In *Proceedings of DARPA Broadcast News Workshop*.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean, 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Ratnaparkhi, Adwait, 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Stroński, Krzysztof and Joanna Tokaj, 2015. The diachrony of cosubordination – lessons from Indo-Aryan. *Proceedings of the 31st South Asian Languages Analysis Roundtable (SALA-31)*:59 – 62. Extended abstract available at: http://ucrel.lancs.ac.uk/sala-31/doc/ABSTRACTBOOK-maincontent.pdf.

Toutanova, Kristina, Dan Klein, Christopher D. Manning, and Yoram Singer, 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of HLT-NAACL*.

Toutanova, Kristina and Christopher D. Manning, 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000)*. Hong Kong.

Van Valin, Robert D. and Randy J. LaPolla, 1997. *Syntax: Structure, Meaning, and Function*. Cambridge: Cambridge University Press.

Van Valin, Robert Jr., 1993. A synopsis of role and reference grammar. *Advances in Role and Reference Grammar*:1 – 164.

Van Valin, Robert Jr., 2005. *Exploring the Syntax-Semantics Interface*. Cambridge University Press.