Hard Problems of Tagset Conversion

Abstract

Part-of-speech or morphological tags are important means of annotation in a vast number of corpora. However, different sets of tags are used in different corpora, even for the same language. Tagset conversion is difficult, and solutions tend to be tailored to a particular pair of tagsets. We discuss Interset, a universal approach that makes the conversion tools reusable. While some morphosyntactic categories are clearly defined and easily ported from one tagset to another, there are also phenomena that are difficult to deal with because of overlapping concepts. In the present paper we focus on some of such problems, discuss their coverage in selected tagsets and propose solutions to unify the respective tagsets' approaches.

1 Introduction

Most annotated corpora use various types of tags to encode additional information on words. In some cases this information is merely the part of speech ("noun", "verb" etc.—hence the term *part-of-speech* or *POS tags*). In many cases, however, the string of characters comprising the tag is a compressed representation of a feature-value structure. Most of the features encoded this way are morphosyntactic (e.g. "gender = masculine", "number = singular"), hence the term *morphological tags*.

Unfortunately, it is very rare to see two corpora sharing a common set of tags. Language differences are only partially responsible—it is the corpus designers, their diverse views, theories and intended uses of the corpora, what matters most. Even two corpora of the same language may define two completely incompatible tagsets.

Such diversity proves disadvantageous for both human users and NLP software. A human user (linguist) typically wants to submit queries such as "show me all occurrences of a noun in plural, preceded by a preposition". Tags however rarely contain statements like "number = plural" literally. That would be prohibitively space-consuming. Instead we have to know that e.g. the fourth character of the tag being "P" means "plural". For instance, the tag `NNIS7-----A----`[1] may read as "part of speech = noun, detailed part of speech = common noun, gender = masculine inanimate, number = singular, case = 7th (instrumental), negativeness = affirmative". To work with the corpus efficiently, a linguist either needs to interpret the tags using specialized software, or to memorize the particular tag scheme. Obviously, if the same linguist has to switch to a different corpus, he/she must memorize more schemes or replace the tag interpretation software.

Similarly, various NLP tools may depend on particular tagsets. While some tools indeed treat tags as atomic strings, others could exploit the tag structure to dig more information about the word—no matter whether they use the features in machine learning, or in human-designed rules. If the tagset changes, manual rules become useless and statistical models have to be retrained at least; even that may not be possible in case the training procedure works with selected subsets of the feature pool. Applicability of NLP software to multiple corpora is exactly the reason why one would want to convert tags from one tagset to another.

For many tagset pairs, designing the conversion procedure is not easy. On one hand, there are rare tagsets (e.g. MULTEXT-EAST, Erjavec (2004)) fitting at the same time languages as distant as Czech and Estonian; on the other hand, tagsets of two closely related languages (e.g. Danish and Swedish) or even two tagsets of the same language may differ substantially (for instance,

[1] This example is taken from the Prague Dependency Treebank (Böhmová et al., 2003).

the Mamba tagset of Swedish (Nivre et al., 2006) contains detailed classification of auxiliary verbs and punctuation but lacks features like number, mood, tense etc.; this is in sharp contrast to another Swedish tagset, Parole (Cinková and Pomikálek, 2006), which in turn is not compatible with the Danish Parole (Kromann et al., 2004) tagset (the former classifies participles as verb forms, the latter as adjective forms; the former has separate tags for numerals, the latter classifies both cardinal and ordinal numbers as adjectives; etc.)

From the above said it follows that the typical tag conversion is an information-losing process. Though it is often desirable to perform it anyway and preserve as much information as possible. Creation of a conversion procedure between two tagsets requires hours of tedious work, consisting mostly of reading the tagging guidelines and translating them into a programming language. A universal description, to which all tagsets map, could make this process easier, and its results reusable. One attempt to find such description and deploy it in the conversion task is DZ Interset (XXXXX, 2008b). In the present paper we discuss the development of the universal description and focus on selected hard problems that arise when comparing various existing tagsets.

The rest of the paper is organized as follows: In Section 2 we describe Interset and how it works. In Section 2.3 we describe our universal set of features. Section 3 gives some implementation details of Interset-based conversion. Then, Section 4 lists decisions that are difficult w.r.t. universality, demonstrates them on real tagsets, and proposes solutions.

2 Interset

Interset is a universal set of features and their values. It shall be able to store all features that are usually encoded in tags. The role of this universal set is similar to the role of Interlingua in Interlingua-based machine translation (Richens, 1958) or the role of Unicode among character sets. The Interset serves as an intermediate step on the way from tagset A to tagset B. The interaction between the Interset and tagsets A and B, respectively, is described in *tagset drivers*. Once the drivers have been implemented, we can do the two-way conversion A to B and B to A, plus the conversion between one of these tagsets and any other tagset that has been defined so far.

We are not likely to spare much effort during the initial phase, if compared to just writing a targeted A-to-B conversion procedure. Actually, covering two completely new tagsets requires more work and care: we should describe both encoding and decoding of each tagset, we may have to think about features that are present in neither of them, and we will probably want to be more careful about aspects that may not matter to our current application. However, the reusability of the resulting code should compensate for the effort more than adequately; it is even possible that the required tagset has been covered by someone else who is sharing the code on the web.

Besides abstract concept definitions, Interset also comprises some real software—supporting procedures that make adding new tagsets easier. Thanks to the encoding algorithm implemented in the support library, developers adding new tagsets need not (not necessarily) consider all features that are irrelevant to the tagset being added (but which may become relevant during conversion to a particular other tagset).

2.1 A New Standard?

Interset is not a new annotation standard. There have been attempts to standardize morphosyntactic tagging and it is not Interset's mission to compete with them. Instead, the goal is to cover as many existing tagsets as possible whether they conform to a standard or not. The set of Interset features and values could of course be compared to those defined in standards. There have been several European projects concerning tagset standardization. The EAGLES project (EAGLES, 1996; Leech and Wilson, 1999) produced a set of recommendations for tagsets. Output of the LE-PAROLE project (Volz and Lenz, 1996) was a multilingual corpus of 14 European languages, morphosyntactically annotated according to a common core PAROLE tagset, extended with a set of language specific features. Another multilingual corpus with common tagset is MULTEXT (Ide and Véronis, 1994) for six European languages (English, French, Spanish, German, Italian and Dutch), and later its spin-off MULTEXT-EAST (Erjavec, 2004) for 12 languages (English, Bulgarian, Czech, Estonian, Hungarian, Romanian, Slovene, and in later versions also Croatian, Lithuanian, Resian, Russian and Serbian); the tagsets used in MULTEXT corpora comply with EAGLES. Various EAGLES-compliant tagsets can be added to our system and

their mutual similarity will probably make adding them all easier. Weakly related is also the Gold Ontology project (Farrar and Langendoen, 2003) that defines various linguistic concepts, some of which serve as feature names and feature values in Interset. Similarly, morphosyntactic and other terms are included in IsoCat.[2]

Interset has been used in cross-language parser adaptation (XXXXX, 2008a), and in MorphCon, a GUI program that brings tag conversion to corpus users (Pořízka and Schäfer, 2009). Currently, it is being deployed as a means of unified access to morphosyntactic annotation for the users of the parallel multilingual corpus Intercorp.[3]

2.2 A New Tagset?

Interset is not primarily meant as a new *physical* tagset for annotation. Although it obviously could be used that way (possibly after compressing the feature values), it is better thought of as a set of concepts that physical tagsets map to. Design of physical tagsets is often guided by various needs such as conforming to linguistic tradition and terminology of the given language, minimizing errors of automatic disambiguation etc. In contrast, the most important design constraint for Interset is the portability of information from one tagset to the others. If a feature value X, encoded in tagset A, is not available in tagset B but it is still probable that users of tagset B would tag the same set of words with feature value Y, then it is desirable that the Interset algorithms are able to change X to Y, when converting from A to B. At the same time, converting to tagset C might require changing X to Z, converting to tagset D would keep X (because D has X, too) and converting to other tagsets would result in resetting X to empty value because there is no better choice available.

Conversion via Interset often looses information but never adds new information. Interset may define feature value X but it just won't be set unless the source tagset defines it, too. Specifically, the conversion procedure does not retag words. For instance, the source tagset may define one tag `IN` for both prepositions and subordinating conjunctions. The target tagset may have separate tags for each of those categories. Will then Interset sort out the words tagged `IN` into prepositions and conjunctions? No! In fact, the procedure *never* looks at the word the tag is assigned to. It only works with the tag itself.

2.3 Features

The current version of Interset defines the following features and values. Besides the values listed, every feature is also allowed to have an empty value.

- **pos** = noun|adj|num|verb|adv|prep|conj|part|int|punc
- **subpos** = prop|class|pdt|det|art|aux|cop|mod|ex|voc|post|circ|preppron|comprep|coor|sub|comp|emp|res|inf|vbp
- **prontype** = prs|rcp|int|rel|dem|neg|ind|tot
- **numtype** = card|ord|mult|frac|gen
- **numform** = word|digit|roman
- **numvalue** = 1|2|3
- **advtype** = man|loc|tim|deg|cau
- **punctype** = peri|qest|excl|quot|brck|comm|colo|semi|dash|symb|root
- **puncside** = ini|fin
- **synpos** = subst|attr|adv|pred
- **poss** = poss
- **reflex** = reflex
- **negativeness** = pos|neg
- **definiteness** = ind|def|red
- **foreign** = foreign
- **gender** = masc|fem|com|neut
- **possgender** = masc|fem|com|neut
- **animateness** = anim|nhum|inan
- **number** = sing|dual|plu
- **case** = nom|gen|dat|acc|voc|loc|ins|ill|ine|ela|all|ade|abl|par|tmp|ter|tra|ess|abe|com|cau|dis
- **prepcase** = npr|pre
- **degree** = pos|comp|sub|abs
- **person** = 1|2|3
- **politeness** = inf|pol
- **subcat** = intr|tran
- **verbform** = fin|inf|sup|part|trans|ger
- **mood** = ind|imp|cnd|sub|jus
- **tense** = past|pres|fut
- **subtense** = aor|imp|pqp
- **aspect** = imp|perf
- **voice** = act|pass
- **abbr** = abbr
- **hyph** = hyph
- **style** = arch|form|norm|coll
- **typo** = typo
- **variant** = short|long|0|1|2|3|4|5|6|

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

[3] http://www.korpus.cz/intercorp/?req=doc:uvod

7|8|9

- **other** = *any other info*
- **tagset** = *where does the other info come from?*

The other feature can store arbitrary information that did not fit elsewhere. Typically, that sort of information is unique to the given tagset (not just unique to the language!), and goes too far from what tagsets usually encode. We do not believe that there will ever be another tagset encoding such information—if we did, we would update Interset to provide corresponding feature/value. However, it still makes sense to store the information so that a tagset can be converted to itself and no information is lost. Since the value of other is not understood by any other tagset, we need to know which tagset the value comes from. Thus the identifier of the tagset should be stored in the tagset feature.

While several feature-based tagsets distinguish between unknown values and irrelevant features, we do not find it wise in Interset. For instance, the fifth character in the PDT Czech tagset identifies grammatical case. Its normal values are 1 to 7. For parts of speech that do not have case (e.g. interjections) the fifth character is - (dash). Adjectives generally do have case, yet there are borrowed words without Czech case suffixes whose case value is unknown (X). An example is the tag AAIPX----1A---- for "Buenos" in Buenos Aires. The benefit of making this distinction explicit in a tagset is unclear. What is clear, however, is that we must not reflect it in the universal feature set. Who can say that a feature will be irrelevant—given the context of the values of the other features—in any tagset whatsoever? It is quite easy to find features that are relevant in one tagset and not the other: e.g. Czech past participles distinguish gender, English don't.

3 Tagset Drivers

While the Interset is merely an abstract definition, the real implementation lies in the tagset drivers. A driver is a code library responsible for decoding and encoding tags. Decoding is reading a string (tag) into an internal data structure, in accordance with the list of possible features and their values. Encoding works the other way around.

The encoder obviously is the more difficult part. The decoder just reads and sorts the information, ideally not losing a single piece of it. If anything has to be discarded because it does not fit the target tagset, the discarding is encoder's task. There are two main reasons why encoding is not easy:

- The encoder should be prepared to all values of all features, regardless that some of them are unknown in the particular tagset. For instance, if number = dual and the tagset does not know dual, it is probably better to encode plural than just leave number unknown.
- Even if the target tagset knows features A and B, concrete value of A can restrict permitted values of B. Some combinations of feature values are not allowed. For instance, the Swedish Parole tagset allows "pos = noun & gender = common | neuter", and also "pos = pronoun & gender = masculine | feminine | common | neuter". If we are to encode "pos = noun & gender = masculine", we can either honor the part of speech, or the gender, but not both.

Fortunately enough, unknown feature values / combinations can be dealt with automatically if the driver has the list of all possible tags. By decoding all tags on the list, we get feature values for every tag. We thus know all feature values permitted in the given tagset and we know all value combinations. We have defined an ordered list of back-off values for every Interset feature value. The back-off lists contain all other values of the feature, including the empty value, so it is guaranteed that we always find a value that is permitted.[4] Of course, the encoder can override the default back-off list if necessary.

As for unknown feature combinations, there is a predefined total ordering of the features that defines their priority (this can be overridden, too). Since features are ordered, all value combinations can be stored in a trie structure. On selecting value of a higher-priority feature, the structure immediately reveals restricted value space for all lower-priority features.

This back-off technique is implemented in a helper module. Any driver can call it and have the features adjusted to something the driver itself might produce during decoding. The encoder can then concentrate on the driver's native feature combinations. Besides that, the helper module can also check a driver's integrity by looking whether the decoder only sets known features and values, whether encode(decode(x)) = x etc.

[4] The necessary condition is that the decoder only sets known feature values, which is desirable anyway.

The whole thing is implemented in Perl. The drivers are Perl modules whose encode and decode functions can be called from other Perl programs, either to access the feature values, or to convert tagsets. The conversion script is very simple and looks like this:[5]

```
use tagset::cs::pdt;
use tagset::en::penn;
while(<>)
{
    print tagset::en::penn::encode
        tagset::cs::pdt::decode $_, "\n";
}
```

At the time of writing there are drivers for 20 tagsets of 10 languages (Arabic, Bulgarian, Chinese, Czech, Danish, English, German, Polish, Portuguese and Swedish). Those drivers are freely available on the web.[6] We believe that the reusability will only be truly exploited if the drivers are shared in the community and we encourage everyone to contribute with drivers they need to write for themselves.

4 The Hard Problems

Working with various tagsets, we identified several fields that were difficult to capture and unify.

4.1 Pronouns, Determiners, Articles and Wh-Adverbs

Most languages and tagsets have personal pronouns, i.e. words like "I", "you", "he", "she", "it". Various interrogative and relative function words (wh-words in English) are often also considered pronouns. In addition, grammars of some languages distinguish *determiners* while others prefer to categorize the same thing as a sort of pronouns. According to the definition of EAGLES, **pronoun** is a function word that *replaces* a noun phrase, while **determiner** is a function word that *modifies* a noun phrase. As a result, proper EAGLES-pronouns behave like nouns and determiners behave like adjectives. Note that possessive pronouns (i.e. "my", "your", "his", "her", "its"), also found in many languages, are personal possessive determiners in the sense of the EAGLES definition.

[5] Real conversion script would also have to deal with the format in which the tags are mixed with text in the corpus. This example merely assumes a list of tags, without the actual words and other annotation.

[6] `http://xxxxx.xx/`

Because tagsets often disagree in what is pronoun, what is determiner etc., it is difficult to find a unifying approach. We decided to limit the number of the major parts of speech in order to minimize the cases where a word would end up with an empty part of speech. If there were a part of speech called *determiner*, drivers of tagsets not having determiners would either have to check whether **pos** = `det` during encoding, or they would fall back into a residual word class. On the other hand, if we tag determiners as special cases of adjectives (which is what Interset does), such drivers will simply encode determiners as adjectives (provided they have adjectives—but these are much more common).

We also followed this solution with pronouns because of the following reasons:

- Although pronouns are found in most tagsets, there is much controversy about the precise extent of that category.
- Some tagsets allow for distinguishing between *substantive* and *attributive* pronouns (roughly those that behave like nouns and those that behave like adjectives). Assigning pronouns to nouns and adjectives respectively helps preserve that distinction.
- Some of the features that distinguish pronouns from real nouns and adjectives (interrogativeness, for instance) are found with adverbs and numerals as well and thus it makes sense to separate them.

Pronouns are recognized by nonempty value of the `prontype` feature; default type is `prs`, which denotes personal pronouns under nouns, and possessive pronouns under adjectives. The drawback of this approach is that the encoding procedure of a driver that recognizes pronouns must define its own method of asking the question *"Does the current feature-value structure correspond to a pronoun?"*

4.2 Numerals

Numerals form an analogy to the pronoun-determiner problem. Most tagsets recognize cardinal numbers as a separate category. However, the rest is less obvious. Some tagsets recognize ordinal numbers while others classify them as adjectives because of their syntactic behavior. An extreme case is the Danish Parole tagset where all numbers including cardinals are adjectives; however, they form a distinguished subclass of adjectives and can thus be recognized. Tagsets of Czech

and other Slavic languages define complex systems of numerals, according to traditional local grammars: besides cardinals and ordinals, there are separate tags for fractions ("one fifth"), multiplications ("five times"),[7] adverbial ordinals ("for the first time"), various types of generic ordinals ("twofold", "four kinds of"), interrogative or relative numerals ("how many", "what",[8] "how many times"), demonstrative numerals ("that many", "that many times"), indefinite numerals ("few", "much") etc.

For the sake of consistency, the solution for numerals should be parallel to that of pronouns. Cardinal numbers, as the most specific and most widely recognized category, should retain their independence. The rest will be split among nouns (fractions), adjectives (ordinals and some generics) and adverbs (multiplications and ordinal points in time). Non-empty `numtype` feature will distinguish them from real nouns, adjectives and adverbs. Parallelly, `prontype` could be set, too, for interrogative, demonstrative and indefinite numerals.

4.3 Non-Finite Verb Forms

Non-finite verb forms often constitute mixed categories from the syntactic point of view. The syntactic properties of participles overlap with adjectives. Similarly, gerunds resemble nouns and transgressives function as adverbs. At the same time however, they retain their verbal arguments.

Usually, these words are tagged as forms of verbs. However, there are exceptions. In the Swedish Parole tagset, participles are tagged as adjectives. Czech equivalent of gerunds is considered a deverbative noun; Czech participles have long and short forms, the former being tagged as adjectives, the latter as verbs. Some tagsets could even delimit participles as a separate part of speech, without clearly marking their affinity to adjectives or verbs. The Interset guidelines prefer marking all three categories (participles, gerunds and transgressives) as verbs. Note however that any guidelines are only to ensure unified approach to different presentations of the same information. It does not apply to information that simply is not there. If participles were tagged as *normal* adjectives without sub-dividing adjectives into "normal ones" and participles, they would remain so in Interset and also in the target tagset.

4.4 Particles and Other Minor Parts of Speech

Every tagset contains a number of specialized tags for small groups of words. They encode endemic phenomena that do not occur in other corpora/languages or simply are not distinguished there (because they are covered by other, broader-scope tags). Sometimes such tags are grouped as *particles* or various "residual" classes. The approach of Interset is to roof them with some more common parts of speech, instead of introducing a new high-level class. We wanted to reduce the necessity of encoders' taking care of parts of speech unknown in their home tagsets. Roofed word classes are usually distinguishable by one of the detailed-part-of-speech features.

An overview of various sorts of particles follows:

- unclassified particle (Czech `TT`, English `RP`, Swedish `Q`)
- interrogative particle (Arabic `FI` هل *(hal)*, Bulgarian `Tn` *ли (li)*)
- affirmative particle (Bulgarian `Ta` *да (da)*)
- negative particle (Arabic `FN` لا *(lā)*, Bulgarian `Tn` *не (ne)*, German `PTKNEG` *nicht*)
- response particle (German `PTKANT` *ja* = "yes", *nein* = "no", *doch* = "yes", *danke* = "thank you"…)
- auxiliary particle (Bulgarian `Tx` *да (da)* = "to", *ще (šte)* = "will")
- modal particle (Bulgarian `Tm` *май (maj)* = "possibly")
- verbal particle (Bulgarian `Tv` *нека (neka)* = "let")
- emphasis particle (Bulgarian `Te` *даже (daže)* = "even")
- gradable particle (Bulgarian `Tg` *най (naj)* = "most")
- unique POS (Danish `U`, covering the words *at* = infinitival "to", *som*, *der*)
- infinitive mark (German `PTKZU` *zu*, Swedish `IM` *att*, English `TO` *to* – includes prepositional occurrences of *to*)
- separated verbal prefix (German `PTKVZ`, *vor* in *stellen Sie sich vor*)
- adjectival particle (German `PTKA`, *am* in *am besten*, *zu* in *zu groß*)

[7] Note that in these languages such numerals can be expressed in one token, which justifies devising a tag for them.

[8] Here, the expected answer is "first", "second" etc. The "what" corresponds to different Slavic word than in e.g. "What do you do for living?" An example from Czech is *kolikátý*.

- existential *there* in English (EX)
- measure word, quantifier (Chinese DM)
- genitive particle de in Chinese (DE) 的 and 得
- Chinese particles 了 *(le)* (perfect), 著 *(zhe)*, 起 *(qǐ)*, 過 *(guò)* (Di)
- Chinese particles 了 *(le)*, 的 *(de)*, 來 *(lái)* (Ta)
- Chinese particles 而已 *('éryǐ)*, 沒有 *(méiyǒu)*, 也罷 *(yěba)*, 沒有 *(méiyǒu)*, 好了 *(hǎole)* (Tb)
- Chinese particles 呢 *(ne)*, 吧 *(ba)*, 啊 *('a)*, 囉 *(luō)* (Tc)
- Chinese particles 嗎 *(ma)*, 否 *(fǒu)* (Td)

4.5 Alternate Values

Some tagsets contain tags that encode two or more values of the same feature at the same time. For instance, the Czech PDT tags define 11 values for the third character of the tag, which denotes gender and animateness. Four are more or less independent values: M = "masculine animate", I = "masculine inanimate", F = "feminine" and N = "neuter". The rest are various combinations: Y=(M|I), T=(I|F), W=(I|N), H and Q=(F|N), Z=(M|I|N), X=(M|I|F|N). The obvious goal here is to make life easier for taggers because some word forms are systematically identical for two or more genders.

Interset does not define separate feature values for all combinations. However, it does allow for storing alternate values if necessary. Since the feature structure is implemented as a Perl hash, a reference to an array of values can be assigned to any feature in the decoder:

```
elsif($gender eq "H")
{
    $f{gender} = ["fem", "neut"];
}
```

At the first glance, such option poses a serious obstacle for encoder design. Before, the encoder was a nice sequence of if statements, merely saying "if gender is fem, output character F". Do we now need to check whether gender is array first? Even if we are writing driver for a tagset that never works with alternate values? Fortunately, no. Once again, the support library provides a procedure that can convert arrays to single values in all features except those that we acutally expect to contain arrays.

There is nonetheless one open question about alternate values. Interset cannot express permissible combinations of multiple features. In the Czech example above, Q actually means more than just "feminine or neuter". It means "feminine singular or neuter plural". Interset can store both genders and both numbers but by that it will also cover feminine plurals and neuter singulars.

4.6 Joint Categories

Every tagset assumes a certain tokenization of the tagged text. The tokenization guidelines may leave unsplit tokens that evolved historically from two words with separate categories. Examples include German *zum = zu dem* "to the", Czech *proň = pro něj* "for him" or *žes = že jsi* "that you have". So in the German example, we have a token corresponding to two other tokens also occurring in the corpus, one tagged as preposition and the other as article.

Much more difficult is Arabic where orthographic rules dictate not to space-separate certain sequences of words. So for instance, the conjunction و *(wa)* "and", often thrown in at the sentence beginnings, is connected to the following word, as are prepositions. Tokenization cannot be finished before morphological analysis because word segmentation may have ambiguous solutions. Thus, in the following example, we'll end up with long tags CONJ+PREP+NOUN_PROP and NOUN+CASE_DEF_NOM, respectively.

```
<token_Arabic>وبالفالوجة
  <voc>wabiAlfAlwjp</voc>
  <pos>wa/CONJ + bi/PREP +
       AlfAlwjp/NOUN_PROP</pos>
</token_Arabic>
<token_Arabic>مثال
  <voc>mivAlu</voc>
  <pos>mivAl/NOUN + u/CASE_DEF_NOM</pos>
</token_Arabic>
```

Note that the first of the two examples corresponds to a sequence of three words (at least from the English perspective) while the second example describes just one noun (stem + suffix). The latter could be perfectly described in Interset; the former is the problem.

Interset currently covers Arabic tagset of the Prague Arabic Dependency Treebank (Hajič et al., 2004), which is already tokenized. For the more moderate examples from Czech and German, it pretends that one of the joint categories is a special feature of the other. Thus for *zum*, Interset would note that it's a preposition with a special property *attached article*. It would not express any relation to how independent articles are tagged. Alterna-

tively, we could use the array approach described in Section 4.5 and assign two values to the pos feature. Neither solution is optimal because both are far from universal. In theory any part of speech could combine with any other and there could be more than two concatenated tokens, each with its own features that should not be mixed all together. Future versions of Interset might introduce a concept of arrays of sequential features structures (to solve the Arabic puzzle above) and arrays of parallel feature structures (to cover the *permissible feature combinations* discussed in Section 4.5). In the meantime, the conversion program using Interset has to split joint tags using its own means.

5 Conclusion

We have described Interset, a reusable and universal method for tagset conversion via common set of features and their values. We have shown how tags from a source tagset are *decoded* into this intermediate feature space, and how their *encoding* into target tagset can automatically replace unavailable feature values by carefully selected defaults that minimize possible damage. We also briefly compared Interset to tagset standardization efforts and pointed out the differences in goals between standards and Interset. Finally, we presented numerous examples from real tagsets to illustrate the most difficult parts of tagset conversion. The solutions we proposed pursue the ultimate goal that information loss is minimized and similar information is encoded similarly, across tagsets and languages.

Acknowledgements

References

Alena Böhmová, Jan Hajič, Eva Hajičová, and Barbora Hladká. *The Prague Dependency Treebank: A Three-Level Annotation Scenario*, chapter 7, pages 103–128. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003. 1

Silvie Cinková and Jan Pomikálek. Lempas: A make-do lemmatizer for the swedish parole-corpus. *The Prague Bulletin of Mathematical Linguistics*, 86:47–54, 2006. 1

EAGLES. Recommendations for the morphosyntactic annotation of corpora, 1996. URL http://www.ilc.cnr.it/EAGLES96/annotate/annotate.html. 2.1

Tomaž Erjavec. Multext-east version 3: Multilingual morphosyntactic specifications, lexicons and corpora. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*, Lisboa, Portugal, 2004. 1, 2.1

Scott Farrar and D. Terence Langendoen. A linguistic ontology for the semantic web. *GLOT International*, 7(3):97–100, 2003. URL http://www.linguistics-ontology.org/gold.html. 2.1

Jan Hajič, Otakar Smrž, Petr Zemánek, Jan Šnaidauf, and Emanuel Beška. Prague arabic dependency treebank: Development in data and tools. In *Proceedings of NEMLAR-2004*, pages 110–117, 2004. 4.6

Nancy Ide and Jean Véronis. Multext (multilingual tools and corpora). In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, Kyoto, Japan, 1994. URL http://www.aclweb.org/anthology/C/C94/C94-1097.pdf. 2.1

Matthias T. Kromann, Line Mikkelsen, and Stine Kern Lynge. Danish dependency treebank. In *http://www.id.cbs.dk/ mtk/treebank/*, København, Denmark, 2004. 1

Geoffrey Leech and Andrew Wilson. Standards for tagsets. In *Syntactic Wordclass Tagging. Text, Speech and Language Technology*, pages 55–80, Dordrecht, The Netherlands, 1999. Kluwer Academic Publishers. ISBN 0-7923-5896-1. 2.1

Joakim Nivre, Jens Nilsson, and Johan Hall. Talbanken05: A swedish treebank with phrase structure and dependency annotation. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, Genova, Italy, 2006. 1

Petr Pořízka and Markus Schäfer. Morphcon – program pro konverzi českých morfologických tagsetů (morphcon – a program for conversion of czech morphosyntactic tagsets). In *Čeština ve formální gramatice*, Brno, Czechia, February 2009. 2.1

Richard Hook Richens. Interlingual machine translation. *The Computer Journal*, 1(3):144–147, 1958. 2

Norbert Volz and Suzanne Lenz. Multilingual corpus tagset specifications. mlap parole 63–386 wp 4.1.4, 1996. URL http://www.elda.org/catalogue/en/text/doc/parole.html. 2.1

XXXXX. Xxxxx. In *Proceedings of IJCNLP workshop on NLP of less privileged languages (NLPLPL)*, Hyderabad, India, 2008a. 2.1

XXXXX. Xxxxx. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco, 2008b. European Language Resources Association. ISBN 2-9517408-4-0. 1