NeoTag: a POS Tagger for Grammatical Neologism Detection

Maarten Janssen

IULA, Universitat Pompeu Fabra
Barcelona
Maarten.Janssen@upf.edu

Abstract

POS Taggers typically fail to correctly tag grammatical neologisms: for a known word, most taggers will only take known tags into account, and hence discard the possibility that that word is used in a novel or deviant grammatical category in a new text. Grammatical neologisms are relatively rare, and therefore do not pose a significant problem for the overall performance of a tagger. But for studies on neologisms and grammaticalization processes, this makes traditional taggers rather unfit. This article describes a modified POS tagger that explicitly considers new tags for known words, hence making it better fit for neologism research. This tagger, called NeoTag, has an overall accuracy that is comparable to other taggers, but scores much better for grammatical neologisms. To achieve this, the tagger applies a system of *lexical smoothing*, which adds new categories to known words based on known homographs. NeoTag also lemmatizes words as part of the tagging system, achieving a high accuracy on lemmatization for both known and unknown words, without the need for an external lexicon. The use of NeoTag is not restricted to grammatical neologism detection, and it can be used for other purposes as well.

Keywords: Grammatical neologisms, Lemmatization, POS Tagging

1. Introduction

POS Tagging is a task that has been considered successfully completed for a while now, with top-of the line taggers reaching around 98% accuracy. However, POS taggers do not score equally well across the board; their average assignment score is high, but there are areas where they score considerably less. The most radical case is that of grammatical neologisms: words that are used in a different grammatical category than they were before. In terms of POS tagging, this means words that are known in the training corpus, but are used in a different POS in the text to be tagged. Most if not all existing parsers will always get such words wrong, since for words that are in the training corpus, they only try to determine the most likely candidate amongst the tags that are used for that word in the training corpus. This is not typically a significant problem for taggers, since grammatical neologism are sufficiently rare to not merit the complexity that arises from attempting to tag grammatical neologisms correctly. When grammatical neologisms are, however, the objective of study, this is a major drawback of existing POS taggers.

This article describes a tagger, called NeoTag, that was specifically designed to deal with grammatical neologisms in a more reliable way. This tagger was designed as part of the APLE project for neologism research (FFI2009-12188-C05-01), which has the semi-automatic detection of grammatical neologisms as one of its objectives.

Semi-automatic neologism detection is the computer-aided process of extracting neologistic occurrences from a given text, called the *study corpus*. A list of words that could be neologisms, or neologism candidates, is extracted automatically from the study corpus, after which the candidates are manually filtered to separate the real neologisms from the false candidates. Existing tool for semi-automatic neologism detection, such as Cénit (Roche and Bowker, 1999), SEXTAN (Vivaldi, 2000), NeoTrack (Janssen, 2004), and Buscaneo (Cabré and Estopà, 2009), focus only on formal neologisms, that is, words that have not been used before. This article describes how NeoTag can be used as a tool for the detection of grammatical neologism, which are those words that have been used before, but only in a different grammatical category. Although NeoTag was built with a specific purpose in mind, it is a general purpose tagger with several advantages over existing taggers outside the context of neologism detection.

The next section describes the basic set-up of the NeoTag parser itself, as well as its main distinctive features: stochastic lemmatization, lexical smoothing, and lexical confidence scoring. After that, section 3 will discuss NeoTag as a tool for the semi-automatic detection of grammatical neologism candidates. As a byproduct of the search for grammatical neologism candidates, NeoTag also detects inconsistencies in already tagged corpus, as described in 3.2.

2. NeoTag Design

2.1. Basic set-up

NeoTag is a straightforward Viterbi or n-gram tagger (Bahl and Mercer, 1976), written in Perl, to which several features have been added. It is a language-independent, customizable tagger, where the behaviour of the tagger can be adjusted by various command-line arguments.

Like any n-gram tagger, NeoTag calculates the most likely POS tag for each word in a text by calculating the probability P for each tag-assignment or *path* (w_1^n, t_1^n) for a sequence of words $w_1 \ldots w_n$, where each w_x is assigned tag t_x. This probability is calculated as in (1) by multiplying the probability $P(w_x, t_x)$ that the word w_x has the tag t_x (called the lexical probability) with the probability $P_t(t_x, t_{x+1})$ that a tag t_x is followed by a tag t_{x+1} (called the transition probability) for each word in the sequence.

$$P(w_1^n, t_1^n) = P(w_1^{n-1}, t_1^{n-1})P(w_n, t_n)P_t(t_{n-x}^{n-1}, t_n) \quad (1)$$

The lexical and transition probabilities are computed from the frequencies in a training corpus. NeoTag reads these probabilities from two parameter files, one with the frequency of each word/tag pair in the training corpus, and one with the frequency of each POS tag n-gram. The system is set-up in a versatile way, designed to make training easy: when the parameter files are not present, the system will look for a training corpus, and build the parameter files on-the-fly.

The variable x in (1) is the amount of context taken into account for POS tag n-grams. In this article, this variable will be assumed the be 1, which mans that only bigrams are used.

An n-gram tagger needs a strategy to deal with unknown words, since unknown words w have a lexical probability $P(w,t) = 0$ for each possible tag t. For this, NeoTag uses a probability assignment strategy based on the end of the word, similar to the strategy implemented by TreeTagger (Schmid, 1994). In the case of words not evidenced in the training corpus, the system looks at other words ending in the same n letters, and uses the frequency distribution of tags for those words as the input for the lexical probability. When insufficient words ending on the same letter are found, a smaller terminal strings is used, which means a back-off to pure POS frequencies if the ending is completely unknown. When an external lexicon is provided, it is also possible to use the external lexical probabilities for unknown words.

NeoTag has an accuracy rating of just over 97% for Spanish when trained on the IULA Spanish gold standard corpus, and similar scores on other corpora that were tested. This score is reached using NeoTag as a bigram tagger without the introduction of any language-specific adjustments, without the use of an external lexicon, and with a training corpus of around 400.000 words. Section 3.2. will show that not all the errors made by the tagger are really errors. A break-down of the accuracy of NeoTag can be found online (http://marke.upf.edu/neotag), but for this article, it is sufficient to say that it reaches an accuracy score similar to other modern POS taggers.

NeoTag is not a compiled and optimized application, but rather a machine-independent Perl script. Nevertheless, NeoTag is sufficiently fast for most purposes, and most importantly, it is fast enough for the purpose of semi-automatic neologism detection. The actual speed of course depends on a lot of features, including the speed of the computer, the size of the training corpus and the tagset, and the parameters used to run the script, most crucially the context parameter x mentioned above. But in the test setting above, it parses over 10,000 tokens per second. Since most study corpora used in semi-automatic neologism detection are small, typically a single day of an online newspaper which rarely amounts to more than 50.000 words, this means tagging is done in a matter of seconds.

2.2. Stochastic Lemmatization

Lemmatization is the task of providing a lemma or citation form for each word in the corpus. This is not a task that POS tagger typically focus on, even though most taggers can provide lemma. For instance, TreeTagger can lemmatize known words (either from the training corpus or from an external lexicon), by looking up the known lemma for the word/tag pair. When the word is not known, taggers typically either do not provide a lemma, or can render the word-form itself as citation form when so asked. This means that with a large enough corpus and/or a large enough lexicon, most words will get lemmatized, but unknown words are not handled. Since neologisms are new, and hence unknown words, this means that neologisms will not get lemmatized, whereas a lemmatized form of neologisms is often useful in neologism research.

NeoTag lemmatizes known and unknown words alike during the tagging process. When it encounters an unknown word, or an unknown word/tag pair, NeoTag looks for known words ending in the same n letters. and checks what their citation form is, that is to say, it checks how the citation form can be (back) formed out of the inflected form. It then lemmatizes the unknown word like the majority of similar words. When no other words ending in the same n letters, the system checks for words ending in the same $n-1$ letters. The amount of letters n can be set by a command line option.

How this works is best shown by an example: when looking at a new unknown word *bewasses*, the system will check for known words ending in *–sses*. With respect to the tag, the training corpus shows that words on *–sses* can be singular nouns, 3rd person indicative verbs, or (most frequently) plural nouns. Once the tagger has establishes by the context that it is, say, a plural noun, NeoTag will attempt to lemmatize *bewasses*. The training corpus shows that there are three possible ways to lemmatize it. Firstly, it could be the plural of *bewas*, similar to *gasses/gas*, or it could be the plural of *bewasse* like *impasses/impasse*. However, the most likely citation form is *bewass*, since the majority of plural nouns in *-sses* follow the pattern *glasses/glass*.

The lemmatization algorithm basically generates a morphological analysis rule on-the-fly by checking how to modify the word-form to obtain the citation form. This modification rule is string-final in NeoTag, meaning that when the inflected form differs from the citation form too far from the end of the word, the system will fail to lemmatize. This does not only happen in language that are not right-inflecting, such as Bantu languages, but also in specific cases in languages that are mostly right-inflecting. For instance, the system is not capable of lemmatizing a Dutch circumfixing past participle like *gedroogde* (dried) to its citation form *drogen*, nor will it manage to lemmatize a left-inflecting portuguese compound like *flores-de-lis* (lilies) to its citation form *flor-de-lis*. However, in the languages on which NeoTag was tested, the amount of cases in which the system fails to lemmatize unknown words is very limited.

When there is only one option for the lemma, the system will simply display the citation form it found. When there is more than one, the system can either select the most likely one, or provide each option, with its respective probability. So in the case of *bewasses*, the option are to either output *bewass* as its lemma, or output *bewass*, *bewas*, and *bewasse* in descending order of likelihood. Only when the tagger fails to lemmatize, does NeoTag revert to outputting an unknown lemma.

This method of lemmatization, without the use of any language specific information or data from outside the training corpus, correctly lemmatizes in over 95% of the cases in our Spanish test. The number of errors in the lemma provided for neologisms is even lower, since neologisms tend to inflect regularly. A significant part of the lemmatization errors are not really errors, but due to inconsistencies in the training corpus (see section 3.2.)

2.3. Lexical Smoothing

Smoothing is a common technique in taggers, and is used to counter problems with data sparseness. It comes traditionally in two types: transition smoothing, and lexical probability smoothing. Transition smoothing is meant to counter unknown sequences of tags, and can be done for instance by Katz back-off smoothing (Katz, 1987). Lexical probability smoothing is used to deal with unknown words, and is done for instance by the word-end mechanism described in section 2.1.

What is needed in the case of grammatical neologism, however, is a different kind of smoothing, which I here call *lexical smoothing*: smoothing the lexical probabilities even for known words, to force the system to consider POS tags for known words that were not evidenced by the training corpus. This is required to account for cases of grammatical neologisms, since grammatical neologisms are known words that are used in a different grammatical category than they were before.

When naively implemented, lexical smoothing blows up the complexity of the tagging task, since every word in the text is considered to possible have any of the tags in the tagset, though be it with a very low probability for most of them. This makes tagging much more complex, and hence much slower (also due to the write-out principle described in the next section). A simple way to implement a more restricted, and hence less costly, version of lexical smoothing is by only considering smoothing for certain tags, for instance by saying that grammatical neologisms can only be nouns. There are two problems with that approach, however: firstly, it means introducing language and tag-set specific characteristics to the system, making it less versatile. And secondly, it is a largely unmotivated restriction on smoothing which does not necessarily smooth in the right places.

NeoTag therefore uses prior evidence as input for smoothing. There are two ways of doing this: either by smoothing known words with the same technique as used for unknown words (word-end smoothing in the case of NeoTag), or by looking at existing cases of cross-category homographs. NeoTag uses the latter approach since it proved to work better for grammatical neologisms. As an example of how this works in NeoTag, consider that in a training corpus, the word *hammer* is used both with the tag N5S and the tag VInf. This is an indication that there is a word tagged as N5S that could have been tagged Vinf as well (in a different context), and that therefore, other N5S might be VInf as well. The more homographs there are for a given pair of tags, the more likely it is that other words of these classes are homographs too. This homographic evidence is then used as a smoothing factor, scaled in such a way that an actual occurrence of a word in a given tag always scores higher than a word/tag pair that is considerd due to lexical smoothing.

Word/tag pairs that are considered due to lexical smoothing are not added to the lexical probability parameter file, but rather created on-the-fly, since treating them as regular lexical probabilities will recursively make them the input of further smoothing steps. Also, lexical smoothing is only used for known words, since in the case of unknown words, the word-end strategy already introduces a sufficient amount of smoothing. Exactly how the smoothing is weighted can be customized by a command-line option, and the optimal amount of smoothing depends on the size of the training corpus, the size of the tag set, and the morphological properties of the language. On the command-line it is also possible to set a threshold to the amount of homographic evidence needed for a pair to be used to smoothing, since otherwise with large training corpus, too many pairs will be considered.

Lexical smoothing corrects cases where, in a sense, the wrong evidence is provided by the training corpus, and as such improves the accuracy of the tagger. However, at the same time it sometimes introduces errors for words that are used with the same grammatical class as in the training corpus, but in an atypical context. As a result, the accuracy of the tagger is in practice often affected negatively when lexical smoothing is done too freely. With the smoothing algorithm of NeoTag, the accuracy is unaffected by smoothing: there is no significant difference in the assignment score of NeoTag with or without lexical smoothing in the training corpora we tested. However, the place where the errors occur changes, making the tagger more accurate for certain application, and especially for the purpose of (grammatical) neologism detection.

2.4. Lexical Confidence

NeoTag calculates path likelihood, but outputs tags based on their lexical confidence score. In a typical bigram tagger, each word gets assigned the tag it has in the most likely path (w_1^n, t_1^n). NeoTag, on the other hand, sums the probability that w_x has a tag t in every path in which w_x has that tag, to calculate the lexical confidence as in (2) and assigns each word the tag with the highest lexical confidence.

$$LC(w_x, t) = \sum_{\{t_1^n | t_x = t\}} P(w_1^n, t_1^n) \tag{2}$$

In practice, the tag assigned to a word w_x in the most likely path hardly ever differs from the tag for that word with the highest lexical confidence score. However, the two are not necessarily identical. In each trial we ran, whenever the two differ, it is *always* the tag with the highest confidence score that is correct, and never the tag from the most likely path. Although the gain in overall accuracy of lexical confidence is small since it only rarely gives a different output, the use of lexical confidence has additional advantages.

One advantage of lexical confidence scores is that they can be given as optional output of the tagger: NeoTag can provide, for each word in the corpus, the lexical confidence score for each possible tag it considered. This gives a quick view on how certain the tag assignment is for a given word

in the corpus, and how many other likely tag candidates there are for that word. Lexical confidence can also be used for detecting potentially problems tag pairs (see section 3.2.). Another advantage of lexical confidence is that it can be used for *feature-backoff*, described in 2.5.

N-gram parsing is a task with optimal substructures in the sense that if there are two paths for a string of words w_1^n that assign the same tag to w_n, then the most likely of those two tag assignments will stay more likely than the other for any longer string of words w_1^{n+x}. This effectively means that to determine the most likely path, we can forget any sub-optimal subpath, and that hence at any point in parsing a corpus, only as many possible paths have to be taken into account as there are possible tags for the last word that was parsed. This also means that whenever a word is encountered for which there is only one possible tag in the training corpus, all tags calculated thus far become certain, and can be written to the output (the write-out principle). Since overzealous lexical smoothing gets rid of words with only one possible tag, this is one of the reasons why too much lexical smoothing makes tagging so much slower.

Lexical confidence scores sum over all paths, including sub-optimal paths. This in principle means that non-optimal paths become relevant as well, which would make parsing much more costly. However, it is possible to circumvent this, by calculating the partial lexical confidence score LC* for each assignment (w_x, t) relative to the tag assigned to the last word in a sequence w_1^n.

$$LC^*(w_x, t|t') = \sum_{\{t_1^n | t_x = t \wedge t_n = t'\}} P(w_1^n, t_1^n) \qquad (3)$$

LC* can be dynamically recalculated with each new word that is parsed, and only needs to be calculated for optimal paths (since all sub-optimal paths have the same last tag as their optimal counterpart). On write-out, it trivially holds that LC=LC*, since each path will have the same assignment for the last word. In this way, lexical confidence scores can be calculating without keeping track of sub-optimal paths.

2.5. Feature Back-off

The lexical confidence scores for a word can be used for a feature I call *feature-backoff*, which comes down to the assignment of a partial tag in uncertain contexts. How this works is best made clear by an example: if the tagger tries to tag a word that can be either a feminine noun or a masculine noun, the system can assign it a noun without a gender marking whenever the context provides too little evidence for one or the other. For instance, the Spanish word *estudiante* (student) can be a masculine or a feminine noun, and in many cases, the context will determine the gender, as in sentence (3), where *estudiantes* has to be feminine. In some contexts, however, as in (4), it is unclear by the given context whether it is masculine or feminine.

Hay estudiantes estupidas en esta clase. (4)

*There are stupid.*FEM *students in this class.*

Hay estudiantes inteligentes en esta clase. (5)

There are intelligent students in this class.

Most taggers mark words like *estudiante* as "gender indifferent" in the lexicon, partially because of contexts like (5). Although this is a practicable solution in many circumstances, it is a sub-optimal solution in others, since the tagger will fail to assign a gender in a sentence like (4), where it is necessarily a feminine noun. This is especially unfortunate for grammatical neologism detection, since gender change is a common source of grammatical neologism.

By comparing lexical confidence scores, NeoTag is capable of dealing with words that can be disambiguated in some contexts, but not in other. For this, the tags need to explicitly indicate masculine/feminine as possible values for the same feature: N[gen=masc][num=plur] vs. N[gen=fem][num=plur] . In a sentence where the context disambiguates as in (3), the LC for the feminine tag will be high, and for the masculine tag low or zero. Yet in ambiguous contexts as in (4), both tags will be comparably high. In those cases, the system can optionally "neutralize" the feature in question: N[gen=masc/fem][num=plur].

Although feature neutralization is fully functional in NeoTag, we have as of yet been unable to test how accurate it is, since we have not been able thus far to compile a large enough corpus with a rich, feature-based tag set. Also, because counting errors has to be done differently because of the possibility of partial matches, the accuracy results are not comparable to existing parsers.

3. Grammatical Neologisms

As said before, NeoTag is meant as a tool for the semi-automatic detection of grammatical neologisms in a given body of text (the study corpus). In principle, NeoTag is not limited to grammatical neologism, but can detect formal neologisms as well: when running NeoTag over a corpus, it not only marks off grammatical neologism candidates, but it also more trivially detects formal neologism candidates. Formal neologism candidates are those words in the corpus that were not part of the training corpus (or the external lexicon), or simply the out-of-vocabulary (OOV) words in the study corpus. This article will concentrate only on NeoTag as a tool for the detection of grammatical neologism. The detection of formal neologisms will not be discussed since on the one hand, using a POS tagger for the detection of formal neologism is not a novel idea, but rather a methodology already implemented in for instance SEXTAN. The only advantage of using NeoTag for formal neologism detection is that NeoTag lemmatizes the neologism candidates as well. And on the other hand, for formal neologism detection, it tends to be better to use an exclusion-based tool such as NeoTrack (Janssen, 2008) or Buscaneo (Cabré and Estopà, 2009).

3.1. Grammatical Neologism Candidates

While tagging a corpus, NeoTag can (optionally) indicate all those cases in which the selected tag was obtained by means of lexical smoothing by putting a percentage sign in front of the tag. By extracting all words words with a percentage sign from the NeoTag output, one directly obtains a list of all those words in the corpus that were assigned a

tag that deviates from the evidence provided by the training corpus (Deviantly Tagged Words, henceforth DTW). DTW are hence those words in the study corpus for which, given their context, it is considered likely that they belong to a different category than the one(s) with which they were used in the training corpus.

DTW are potentially grammatical neologisms, but not necessarily so. Say we have a word w in the study corpus, that has been tagged with t_1 by NeoTag, but for which all occurrences in the training corpus are tagged with t_2. It can be the case that t_1 is simply the wrong tag, and that w is in fact a t_2 in the study corpus. It is also possible that w is not really a word, but rather a punctuation mark, a proper name, etc, and hence should not be counted as a neologism. It is also possible that w in the training corpus was in fact incorrectly tagged, and should have been tagged with t_2 as well, or that t_2 would have been an alternative, but equally correct tag for w in the training corpus (see section 3.2.). Or it can be the case that (w, t_1) is a perfectly normal word that just happens not to have been used in the training corpus. Only when none of these conditions is met, should (w, t_1) be counted properly as a grammatical neologism.

Not only is it possible that DTW are not grammatical neologism, but there can also be word in the study corpus that are not DTW, but should nevertheless be considered grammatical neologism, at least under a lexicographic criterion for neologisms. The lexicographic criterion says that a neologism is a word that is not (yet) included in a lexicographically controlled list of known words of the language, typically a dictionary. A word (w, t_1) in a study corpus can still be a grammatical neologism, even if it occurs in the training corpus, for instance in those cases where w was incorrectly tagged in the training corpus, or when w was a typographic error in the training corpus, or simply when the training corpus is too recent, and itself contains words that should still be considered neologisms.

Because of these discrepancies, NeoTag is not used directly as the source for grammatical neologism candidates, but the output of NeoTag is first compared against a lexicographically controlled lexicon in order to yield the list of candidates. In our set-up, OSLIN is used as the lexicographic resource. OSLIN (Janssen, 2005) is a lexical database with a rich array of lexical information, but for the purpose of this article, it is only relevant that it is a full-form lexicon with explicit indications for each word in the lexicon in which external lexical resources (dictionaries) that word appears. Because of these source indications, we can control exactly which known words we want to consider non-neologistic.

After tagging the study corpus with NeoTag, all words are checked against OSLIN, that is to say, for each word in the study corpus we verify if that word appears in OSLIN with the indicated grammatical category. Whenever a word is not found in OSLIN, it is considered a formal neologism candidate, and it is considered a grammatical neologism candidate if it is found in OSLIN, but with a different grammatical category. This is independent of whether the word is a DTW or not: DTW that are found in OSLIN are discarded, whereas words that are found only with a different grammatical category are considered candidates even if they are not DTW. Tokens belonging to categories that are not considered words are also discarded. Furthermore, it is possible to exclude candidates that result from ambivalent tag pairs, as explained in the next section.

3.2. Corpus Inconsistencies

Because of the use of lexical smoothing, NeoTag will suggest tags that go against the evidence found in the training corpus. That means that if we train NeoTag using a training corpus, and then run NeoTag over that same training corpus, the output will not always match the input. When testing the accuracy of a tagger, the differences between the output of the tagger and the tags provided by the training corpus are considered errors. However, since no training corpus is without problems, the tagger can in principle also indicate corrections in the gold standard corpus. This is true for any tagger, but more so for NeoTag due to the lexical smoothing: because of lexical smoothing, NeoTag considers whether alternative tags would not have been more probable than the one(s) given in the training corpus. Therefore, NeoTag can be used to check the consistency of a gold standard corpus.

Mismatches between the output of NeoTag and the training corpus can be simple cases of errors in the training corpus, for instance where a word was marked as an adjective in the training corpus, whereas it was in fact a noun. Such errors are most often individual errors that would be picked up by any parser, but can be DTW as well. For instance, in one of the corpora that was used for testing, the word *during* was consistently tagged as an adverb. In many contexts, NeoTag (correctly) suggested that *during* was more likely to be a preposition, even though no occurrences of *during* as a preposition were found in the training corpus.

A larger class of mismatches are the cases where two tags are in practice (virtually) indistinguishable. A well-known example is the fuzzy distinction between past participles and adjectives in English: past participle can be used after the the auxiliary verb *have*, as for instance in example (6), where adjectives cannot be used.

John had boiled/*green two eggs. (6)

In (almost) all other contexts, past participles behave just like any other adjective. This means that in many cases, it becomes difficult even for a human to decide whether to tag a participial form as an adjective, or to tag it as a verb form, and both solutions can often be considered equally correct. Although it would be possible to tag all participial forms that are not behind an auxiliary verb as adjectives, most gold standard corpora tag some participial forms as adjectives and others as verb forms, depending on the meaning of the sentence and on whether or not the participial form is listed as an adjective in the dictionary.

In such cases of structural ambiguities between classes, NeoTag will assign high scores for both tags involved. For instance, NeoTag will mark all participial forms in English as potential adjectives and potential verb forms, independently of how they were tagged in the training corpus. This means that NeoTag can be used to automatically detect classes of words where tags overlap, which you might call *ambivalent tag pairs*. Ambivalent tag pairs are two pairs of pairs that are both considered likely in most or all of there

occurrences, that is to say a tag t_1 is ambivalent with t_2 if whenever t_2 has a high lexical confidence, t_1 has a high lexical confidence as well. Although ambivalent tags are not necessarily bad or avoidable in a corpus, it is important to know which tags in which contexts are ambivalent.

In the same manner, NeoTag can be used to detect inconsistencies in the lemmatization of the corpus. In languages where adjectives inflect, such as Spanish, past participles such as *cocidas* (cooked.FEMPLUR) are sometimes lemmatized to their verbal citation form *cocer* (to cook), and sometimes to their masculine singular participial form (*cocido*). Although there are arguments in favour of both of these options, they should not be mixed, whereas in some corpora they are, especially when participial forms are corrected by hand from adjectives to verb forms without correcting the citation form. When lemmatization is done inconsistently in the corpus, NeoTag will mark all such words as having two possible citation forms.

As mentioned in section 2.3., NeoTag uses known homographs for lexical smoothing, since it gives the best results for neologism detection. However, for using NeoTag as a corpus consistency checker, it is better to use lexical smoothing based on word-ending, which is to say, to use the same strategy for lexical smoothing of known words as for lexical smoothing of unknown words.

Grammatical neologism candidates that are the result of (highly) ambivalent tag pairs are poor candidates at best: when NeoTag suggests that *cocidas* is most likely an adjective, whereas it is only listed as a past participle in OSLIN, then *cocidas* is hardly an (interesting) grammatical neologism candidate since it indicates more a change in perspective than a change in the use of the word. Therefore, in the process of extracting grammatical neologism candidates, it is possible to exclude all candidates that result from such ambivalent tag pairs. For this, it is necessary to first run NeoTag to extract a list of (potential) ambivalent tag pairs, and then use that list (potentially cleaned up by manually) as a filter in the process of neologism detection.

3.3. Neologism Detection Accuracy

Establishing the recall and precision of NeoTag as a tool for semi-automatic neologism detection is not trivial task, since there are too many factors that play are role. There is the label assignment score of the tagger itself, the recall and precision of the words marked by the tagger as DTW, and the recall and precision of the words marked as neologism candidates and/or as grammatical neologism candidates. All these scores are affected by how ambivalent tags are dealt with. Furthermore, these scores get more complicated by the (small) margin of errors in the training corpus and the OSLIN database, and by the fact that there are various definitions of what a (grammatical) neologism is in the first place.

To neverthess give an idea about the quality of the detection process, NeoTag was trained against the IULA Gold Standard corpus for Spanish, after which it was used to extract grammatical neologism candidates from some issues of the *El Pais* newspaper from september 2011. As mentioned before, the tagger has a 97% accuracy under these conditions. The amount of grammatical neologism candidates found per day under these conditions is relatively low: there are about 250 candidates in a single newspaper issue without excluding ambivalent pairs, which is similar to the amount of formal neologism candidates extracted for this newspaper by the Buscaneo tool, meaning that it is a reasonable number of candidates for manual treatment. The number of candidates goes down to around 90 when (some) ambivalent tag pairs and tags resulting from other problems with the training corpus are excluded.

In the raw list of grammatical neologism candidates, there is a 7% error margin resulting from tagging errors, with 93% of the candidates being tagged correctly. When filtering out ambivalent pairs, the accuracy goes down to 70%, which is still a percentage that is sufficiently high for use in semi-automatic neologism detection. However, it should be said that when noun/adjective is considered an ambivalent pair, or when words that are not DTW are excluded, the number of correct neologism candidates goes down considerably. A more detailed analysis of the accuracy of NeoTag for grammatical neologism detection can be found online (http://marke.upf.edu/neotag).

4. Conclusion

As shown in this article, it is possible to detect grammatical neologism candidates automatically from a study corpus by using NeoTag, a POS tagger which uses lexical smoothing to force a search for grammatical neologism candidates. NeoTag tags and lemmatizes known and unknown words alike, with a label assignment score comparable to that of other taggers (around 97%). In the output, it marks words that have a POS tag that results from lexical smoothing, that is, known words in the study corpus used in a different grammatical category.

The output of NeoTag is compared against the OSLIN lexical database to yield a list of grammatical neologisms candidates: words used in a grammatical category different from the one in which they are listed in the dictionary. When used to extract candidates from online newspapers, as is common in neologism observatories, this process yields a limited amount of candidates, which is small enough to be manually processed alongside with the formal neologism candidates studied currently by neologism observatories: some 100 or 250 per day, depending on how they are counted. The majority of candidates corresponds indeed to a grammatical neologism, making NeoTag a practicable tool for the semi-automatic detection of grammatical neologisms.

Apart from its use as a tool for neologism research, NeoTag can also be used as a general-purpose POS tagger, which not only tags, but also lemmatized words. Furthermore, because of the lexical smoothing algorithm, NeoTag can be used to verify and correct a gold standard corpus by using NeoTag on the same corpus that it was training on.

At this moment, we are working on the use of NeoTag with a feature-heavy tag set, which includes tags for semantically oriented distinctions such as mass/count nouns, gradable adjectives, etc. Under normal circumstances, such a semantically oriented tag set is hardly usable in POS tagging, but the feature back-off discussed in section 2.5. means that NeoTag is capable of assigning features only in

those circumstances where they are sufficiently evidenced by the context. When used with such a feature-heavy tag set, NeoTag detects a wider range of grammatical neologisms, including words that are typically considered semantic neologisms. However, at this time, we do not yet have a tagged corpus with such a feature-heavy tag set that is large enough to train and test NeoTag under those conditions.

5. References

L. R. Bahl and R. L Mercer. 1976. Part of speech assignment by a statistical decision algorithm. In *IEEE International Symposium on Information Theory*.

Teresa Cabré and Rosa Estopà. 2009. Trabajar en neologa con un entorno integrado en lnea: la estacin de trabajo obneo. *Revista de Investigacin Lingstica*, pages 17–38.

Joshua Goodman and Stanley Chen. 1998. An empirical study of smoothing techniques for language modeling. Technical report TR-10-98, Harvard University, Cambridge.

Maarten Janssen. 2004. Orthographic neologisms: selection criteria and semi-automatic detection. Unpublished manuscript, availabe at: maarten.jansenweb.net/Papers/neologisms.pdf.

Maarten Janssen. 2005. Open source lexical information network. In *Third International Workshop on Generative Approaches to the Lexicon*, pages 400–401.

Maarten Janssen. 2008. Neotrack: Une analyseur de néologismes en ligne. In *Proceedings of CINEO 2008*, Barcelona.

Slava M. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. In *IEEE Transactions on Acoustics, Speech and Signal Processing*, pages 400–401.

Sorcha Roche and Lynne Bowker. 1999. Cénit: Systme de dtection semi-automatique des nólogismes. *Terminologies Nouvelles*.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, pages 88–89.

Jordi Vivaldi. 2000. Sextan: prototip dun sistema dextracció de neologisms. In Freixa Cabré and Solé, editors, *La Neologia en el tombant de segle*, pages 165–173. IULA, Barcelona.