IJCLA VOL. 4, NO. 2, JUL-DEC 2013, PP. 169–188
RECEIVED 05/08/13 ACCEPTED 10/11/13 FINAL 30/11/13

Syntactic Dependency Based N-grams in Rule Based Automatic English as Second Language Grammar Correction

GRIGORI SIDOROV

Instituto Politécnico Nacional, Mexico

ABSTRACT

In this paper, we present a system for automatic English (L2) grammatical error correction. It participated in ConLL 2013 shared tasks. The system applies a set of simple rules for correction of grammatical errors. In some cases, it uses syntactic n-grams, i.e., n-grams that are constructed in a syntactic metric: namely, by following paths in dependency trees, i.e., there is special procedure that allows obtaining syntactic n-grams. Note that in general case syntactic n-grams permit introducing syntactic information into machine learning methods, because syntactic n-grams have all properties of traditional n-grams. The system is simple, practically does not use additional linguistic resources and was constructed in two months. Due to its simplicity it does not obtain better scores as compared to more sophisticated systems that use many resources, the Internet and machine learning methods, but it can be positioned as a baseline system for the task.

1 Introduction

The dominant paradigm in Computational Linguistics (CL) and Natural Language Processing (NLP) nowadays is based on machine learning methods. Most popular are supervised learning techniques because they obtain better results as compared with unsupervised approaches. The shortcoming of the supervised methods is the necessity of large linguistic data suitable for further application of supervised machine learning algorithms. In practice, it turns into the need of the large manually marked corpora. The problem is even bigger because each CL or NLP task needs a specific corpus marked in a unique manner. So, there should be as many different corpora as there are CL and NLP tasks, such as machine translation, automatic summarization, POS tagging, parsing, various levels of semantic and discourse annotation, etc.

On the other hand, an alternative to machine learning is the paradigm based on usage of human crafted rules. It is not so popular nowadays, though it was dominant until a couple of decades ago (until 90s) (Bolshakov, Gelbukh 2004). In this case, humans instead of annotating corpora are creating rules. It is obvious that for creating rules the humans try to take into account the same phenomena as machine learning algorithms. The current state of the art is that a machine learning algorithm can take into account so many textual (and especially contextual) features at many levels of language at the same time that it outperforms humans (Gelbukh 2013).

Interestingly, a growing interest to rule based approaches is related to a relatively new generative machine learning approaches such as Conditional Random Fields. These approaches use hand-crafted features that usually describe local context. It is known that Conditional Random Fields outperform traditional machine learning on classical tasks such as, for example, POS tagging. So, probably, a new paradigm is emerging that will be based both on machine learning algorithms and manually developed rules.

After this brief discussion about the tendency of use of rules in CL and NLP, let us describe the use of rules in the system presented in this paper. The problem discussed in the paper is related to the problem of automatic correction of grammatical errors of persons who are learning English as the second language (L2). Though various methods have been proposed for detecting and correcting such errors of different kinds: semantic errors (Bolshakov, Gelbukh 2003), malapropisms (Boshakov, Galicia-Haro, Gelbukh 2005), errors in lexical functions (Gelbukh, Kolesnivova 2013), the problem remains very relevant. In particular, this problem was represented in the ConLL 2013 shared task.

This paper describes the system that performs this task using a set of hand crafted rules. Some of these rules are based on the concept of syntactic dependency based n-grams (sn-grams), which we proposed recently (Sidorov, Velasquez, Stamatatos, Gelbukh & Chanona-Hernandez 2012, 2013, 2014; Sidorov 2013).

The proposed set of rules is simple and the whole development cycle of the system began about two months before the task deadline and took approximately only one person-month joint effort in total, which is relatively little effort. So it is not surprising that the system does not present excellent results, but instead due to its simplicity and quick development speed it can be positioned as a base line system for the task.

The rest of the paper is organized as follows. First, in Section 2 we describe the concept of syntactic dependency based n-grams that are used by our system. In Section 3 the ConLL shared task is described. After this in Section 4 we present our system and the rules, which it uses. The obtained scores are discussed in Section 5, and finally in Section 6 conclusions are drawn.

2 Syntactic Dependency Based N-grams

In this section we present briefly syntactic dependency based n-grams (syntactic n-grams, sn-grams). We introduced this concept in our previous works (Sidorov et al. 2012, 2013, 2014; Sidorov 2013). We have shown that application of syntactic n-grams gives better results than the use of traditional n-grams for the task of the authorship attribution. Similar idea was proposed in (Pado, Lapata 2007; Gelbukh 1999), but only as something very specific for certain tasks of syntactic or semantic analysis.

Note that sn-grams are not n-grams constructed using POS tags, as one may suppose just looking at the term. In fact, strictly speaking, it is wrong usage of the word "syntactic" because POS tags represent merely morphological data and syntactic information (context) is used only for disambiguation.

For explaining what for syntactic n-grams are used, we need to remind the reader the concept of the Vector Space Model (VSM). Majority of modern machine learning methods is based on Vector Space Model. The VSM is very versatile and can be used for characterization of any types of objects. General idea of the VSM is that any object in the world can be represented using certain features and these features have sets of possible values, so by choosing a set of features we define a VSM for the selected objects. Each object is represented by a vector of values of the selected features, i.e., it is a point in multidimensional vector space, being features the axes. Note that the features are ordered. Since we are talking about vectors, we can calculate their similarity in a formal way using, for example, the cosine measure. Once the VSM is constructed, all calculi are objective, but its construction is subjective: we can choose any features we like and scale the values in a manner we prefer.

Now, when speaking about texts, the features that are used typically for VSM construction are words or traditional n-grams—word sequences as they appear in texts. Usually, tf-idf values are used as values of these features. These values depend on the word or n-gram frequencies in texts.

There is modern research tendency that consists in reducing dimensions of the VSM using methods such as the Latent Semantic Analysis (LSA). It is possible because sets of vectors are equivalent to matrices, and the LSA is in practice an application of standard matrix processing technique−singular value decomposition (SVD).

The Vector Space Model representation is applied practically in any CL and NLP task with slight modifications.

Main criticism of the Vector Space Model is that it is purely statistical and does not reflect linguistic knowledge.

Our proposal is to introduce syntactic knowledge into the VSM by using other type of features, i.e., instead of traditional n-grams that are just sequences of words at the surface level, we suggest using syntactic n-grams that are obtained using linguistic (syntactic) knowledge, so that they reflect "real" relations between words.

The method of obtaining syntactic n-grams consists in following paths in syntactic tree and taking the words for n-grams in the order of their appearance. Obvious disadvantage of these features is that previous parsing is needed, but nowadays there are many freely available fast reliable parsers for many languages. We use dependency trees, but constituency trees can be used as well, because both types of trees reflect the same syntactic reality (Gelbukh, Calvo, Torres 2005).

In our previous works, we have proposed classification of syntactic n-gram types. Depending on the elements that constitute them, there can be syntactic n-grams of words / lemmas, POS tags, SR tags (names of Syntactic Relations), multiword expressions (Gelbukh, Kolesnikova 2013; Ledeneva, Gelbukh, García-Hernández 2008), and even of characters (Sidorov et al. 2012, 2013, 2014). There also can be mixed sn-grams, for example, one element is a POS tag and the other one is a lexical unit.

On the other hand, in (Sidorov, 2013) we have proposed to differentiate between continuous (non-interrupted path) and non-continuous (path with interruptions or returns) syntactic n-grams. The difference is that in case of continuous n-grams we follow the syntactic path as one continuous line, without interruption (returns, bifurcations), while in case of non-continuous n-grams the syntactic path can have interruptions (returns, bifurcations), and, thus, we can return to the same point in the tree. In the latter case special meta-language for syntactic n-gram representation is needed, because there can appear ambiguities. We proposed very simple meta-language with comas and brackets, which allows resolving the problem of ambiguities. It is clear that continuous syntactic n-grams is a special case of non-continuous sn-grams (with no interruptions/bifurcations/returns).

Now let us give some examples. We will use probably the most linguistically famous phrase by N. Chomsky "*Colorless green ideas sleep furiously*", where the words are used without any sense but the syntactic structure is maintained. Obviously, syntactic n-grams can be extracted from any phrase that we can parse.

Stanford parser produces the following output.

amod(ideas-3, colorless-1)
amod(ideas-3, green-2)
nsubj(sleep-4, ideas-3)
root(ROOT-0, sleep-4)
advmod(sleep-4, furiously-5)

Using this data we can construct the corresponding syntactic tree (Fig. 1) and then extract syntactic n-grams. First, let us consider only continuous (non-interrupted) n-grams. We start from the root and follow the arrows without returns. In case of word bigrams we have:

sleep ideas
ideas green
ideas colorless
sleep furiously

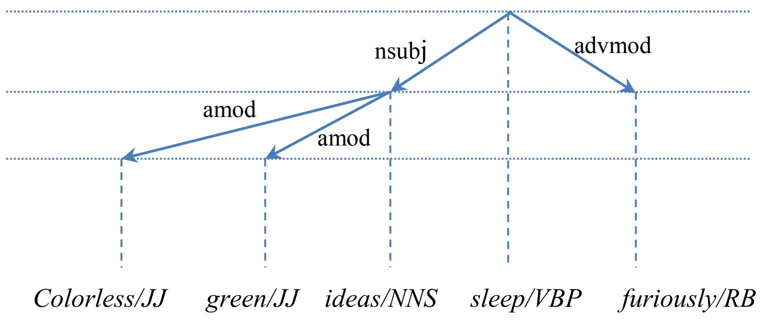

Fig. 1. Example of a dependency syntactic tree

Note that the head word is always the first element of sn-gram. If we compare it with traditional bigrams:

colorless green
green ideas
ideas sleep
sleep furiously

The obvious advantage is that instead of the traditional bigram of two adjectives "*colorless green*" we have the bigram "*ideas colorless*", which has much more sense.

Syntactic 3-grams of words are: *sleep ideas colorless*, *sleep ideas green*.

No more continuous syntactic n-grams of words can be constructed, but our tree is extremely simple. For more complex tree there are much more sn-grams.

We can also consider syntactic n-grams of POS tags, like bigrams *VBP-NNS*, *NNS-JJ*, *NNS-JJ*, *VBP-RB* or trigrams *VBP-NNS-JJ*, *VBP-NNS-JJ*.

Also syntactic n-grams of names of syntactic relations are possible, like *nsubj-amod*. Note that this type of n-grams does not exist for traditional n-grams.

Mixed syntactic n-grams are also possible, for example, if we mix POS tags and words, the following bigrams are extracted: *sleep-NNS*, *ideas-JJ*, *sleep-RB*, *VBP-ideas*, *VBP-furiously*, *NNS-green*, *NNS-colorless*. Also the following 3-grams: *sleep-NNS-JJ*, *sleep-ideas-JJ*.

VBP-NNS-green, *VPB-NNS-colorless*, *VBP-ideas-JJ*, *VBP-ideas-green*, *VBP-ideas-colorless*.

We can also mix SR tags (names of syntactic relations) with POS-tags or words/lemmas, for example, *nsubj-ideas-amod*, *VBP-nsubj-ideas-amod*, etc. In this case, it is a question of future experiments to determine which types of sn-grams give better results.

Now let us pass to non-continuous syntactic n-grams. In our tree, there is only two points of bifurcations: in *sleep* and *ideas*.

Note that in case of bigrams there is no distinction between continuous and non-continuous types.

The rules of the meta-language, which we have proposed for representation of non-continuous sn-grams, are simple: the elements of bifurcation are separated by comas (to distinguish them from a continuous path) and each bifurcation is taken in brackets. The rules are applied recursively. Extraction of these sn-grams can be performed by simple recursive algorithm (Sidorov, 2013).

Non-continuous syntactic 3-grams of words for the example sentence are:

sleep [*ideas*, *furiously*]
ideas [*colorless*, *green*]
sleep ideas colorless
sleep ideas green

There are two more non-continuous 3-grmas as compared to continuous 3-grams, which correspond exactly to bifurcations.

There are three 4-grams in the example.

sleep [*ideas* [*colorless, green*]]
sleep [*ideas colorless, furiously*]
sleep [*ideas green, furiously*]

Note that there is no coma in the first 4-gram between *ideas* and [*colorless, green*], because coma only separates elements of bifurcations.

There is also one non-continuous 5-gram.

sleep [*ideas* [*colorless, green*], *furiously*]

We hope that we have now explained the concept of syntactic n-gram and its types.

```
829 2 1 0    This          DT    1    nsubj   (ROOT(S(NP*)
829 2 1 1    caused        VBD  -1    root    (VP*
829 2 1 2    problem       NN    1    dobj    (NP*)
829 2 1 3    like          IN    1    prep    (PP*
829 2 1 4    the           DT    5    det     (NP(NP*
829 2 1 5    appearance    NN    3    pobj    *)
829 2 1 6    of            IN    5    prep    (PP*
829 2 1 7    slums         NNS   6    pobj    (NP(NP*)
829 2 1 8    which         WDT  16    dobj    (SBAR(WHNP*)
829 2 1 9    most          JJS  16    nsubj   (S(NP(NP*)
829 2 1 10   of            IN    9    prep    (PP*
829 2 1 11   the           DT   12    det     (NP*
829 2 112    time          NN   10    pobj    *)))
829 2 1 13   is            VBZ  16    cop     (VP*
829 2 1 14   not           RB   16    neg     *
829 2 1 15   safe          JJ   16    amod    (ADJP(ADJP*
829 2 1 16   due           JJ    7    rcmod   *)
829 2 1 17   to            TO   16    prep    (PP*
829 2 1 18   the           DT   20    det     (NP*
829 2 1 19   unhealthy     JJ   20    amod    *
829 2 1 20   environment   NN   17    pobj    *))))))))))
829 2 1 21   .             .     -    -       *))
```

Fig. 2. Example of a parsed sentence

3 ConLL Shared Task Description

ConLL Shared Task consists in the following. The training data was available for the registered teams. This data was processed previously by the Stanford parser (de Marneffe, MacCartney, Manning 2006). The data is part of the NUCLE corpus (Dahlmeier, Ng, Wu 2013). The data also contains the error types and the corrections of errors.

For example, the phrase "*This caused problem like the appearance of slums which most of the time is not safe due to the unhealthy environment*" is represented in the parsed variant as shown in Fig. 2.

The first four numbers correspond to the identifiers of the text, paragraph, sentence, and word correspondingly. Then the word itself comes together with its grammar tag (class). Three last columns contain syntactic data. The last column represents the constituency format that

```
<ANNOTATION>
<MISTAKE nid="829" pid="2" sid="1" start_token="2" end_token="3">
<TYPE>Nn</TYPE>
<CORRECTION>problems</CORRECTION>
</MISTAKE>
<MISTAKE nid="829" pid="2" sid="1" start_token="13" end_token="14">
<TYPE>Vform</TYPE>
<CORRECTION>are</CORRECTION>
</MISTAKE>
<MISTAKE nid="829" pid="2" sid="1" start_token="18" end_token="19">
<TYPE>ArtOrDet</TYPE>
<CORRECTION>their</CORRECTION>
</MISTAKE>
</ANNOTATION>
```

Fig. 3. Example of error information.

we, in our case, ignore. The remaining two columns contain the number of the head word (i.e., the word that is the head word for the current one) and the type of the syntactic relation.

The error information is presented in a separate file with XML encoding, see Fig. 3. Information about each error starts with the field <MISTAKE>, where the text, paragraph and sentence IDs are present, while *start_token* and *end_token* indicate position of the error in the sentence. The field <TYPE> contains the error type (see Section 3.1), and the field <CORRECTION> has the suggested correction of the error. For example, there are three errors in the example sentence as shown in Fig. 3. In our opinion, the corpus is a valuable resource because it contains manually annotated data, but it contains many polemic decisions, which can be seen in the example sentence. We would not consider as errors the words marked as the first and the third error. The suggested variants can be slightly preferred, but if they should be considered errors is not so clear.

The subjectivity in corpus preparation no doubt influences the final results of all systems during evaluation. The concept of what is an error should be defined clearer for more precise evaluation. We would suggest that some cases should be marked as “preferred correction” or “possible correction”. Later the systems that do not detect these cases should not be penalized, nor the systems that propose the possible corrections should not have any additional negative score, i.e., neither precision nor recall should be affected. In spite of these shortcomings,

Table 1. Statistics of grammatical errors in the data

Error type	Training data	%	Test data	%
Vform (Verb form)	1,451	9.1	122	7.4
SVA (Subject-verb agreement)	1,529	9.6	124	7.5
ArtOrDet (Article or determiner)	6,654	42.1	690	42.0
Nn (Noun number)	3,773	23.9	396	24.1
Prep (Preposition)	2,402	15.2	311	18.9
Total	15,809		1,643	

the effort of the organizers is valuable and should be highly appreciated.

After the period when the training data is available, the test data in the same format (but without error information) is released. The systems should correct errors in the test data. Special script in Python for evaluation is provided (Dahlmeier and Ng, 2012).

3.1 *Types of Errors Marked in the Data*

There are five types of errors considered in the task: noun number, subject-verb agreement, verb form, article/determiner and choice of prepositions.

Here we present examples of the error types.

First, let us see an example of the subject-verb agreement ("SVA" error type) error. In the phrase "*This endeavor to produce more nuclear power <u>have</u> stimulated the development of safer designs of nuclear reactors*." the auxiliary verb "*have*" should be changed to "*has*".

The other error type is related to use of prepositions ("Prep" error type). The following phrase "*These green house gases are the main cause <u>to</u> worldwide global warming which give rise to further catastrophes such as the rise in global temperature etc*." has an error in the preposition "*to*", which should be substituted by the preposition "*of*".

Error type caused by the wrong usage of a verb form ("Vform" error type) is present in the following sentence. "*Under this process, the attractiveness and practicality of the inventions will be improved such that they could be converted into useful products which <u>accepted</u> by most people*." Instead of the verb form "*accepted*", the form "*are accepted*" should be used.

The other error type is the incorrect choice of an article or a determiner ("ArtOrDet" error type), for example, in "*On one hand more and more virus and hack can access personal computers, so* <u>*the*</u> *secret data and documents may be stolen*." the underlined article should be eliminated.

Finally, the last error is related to the wrong use of noun number ("Nn" error type). In the phrase "*Besides safety, convenience is also desirable for identifications.*" the word "*identification*" should be used in singular.

The errors statistics presented in Table 1 were calculated on the available data.

It can be observed that the test and training data are more or less proportional and the larger percentage of error types are "Article or Determiner" errors, followed by "Noun number" and "Preposition". As usual, during the percentage calculus rounding effects can affect the total percentage value.

4 System Description

The system uses training data for construction of syntactic n-grams only (in this case they are used as syntactic patterns), and then apply a set of simple rules described below trying to detect each one of the five error types one after another in each sentence from the test data and correct them.

Error detection is done in certain order. We first process the possible "Noun number" errors, because if we process them later, then the errors in agreement are produced. If we want to correct these errors, we should also change the corresponding verb as far as its agreement is concerned. Fortunately, as the syntactic information is available, we can easily find the verb-noun (as the subject or part of the predicate) pairs.

4.1 *Linguistic Data Used by the System*

The system uses very few linguistic data, such as word lists, corpora or dictionaries.

First of all, it is necessary to mention that though the morphological data is present in the input sentence (parsed by the

Stanford parser), it is necessary to be able to perform morphological generation for producing the corrections of the errors. For this we need either English list of word forms with corresponding grammatical information and lemmas or algorithms of morphological analysis and generation. We used freely available word list from the FreeLing software (Padró, Collado, Reese, Lloberes, Castellón 2010). The list contains word forms, their grammar tag and the lemma for about 71,000 lemmas. Note that several grammar tags or even lemmas can correspond to the same word form, so the search should take them all into account.

...boarded board VBD board VBN
boarder boarder NN
boarders boarder NNS
boarding board VBG
boardroom boardroom NN
boardrooms boardroom NNS
boards board NNS board VBZ
boars boar NNS
boas boa NNS
boast boast NN boast VB boast VBP
boasted boast VBD boast VBN
boastful boastful JJ
boasting boast VBG
boasts boast NNS boast VBZ
boat boat NN boat VB boat VBP
boatbuilder boatbuilder NN
boatbuilders boatbuilder NNS
boated boat VBD boat VBN
boater boater NN
boaters boater NNS...

This list is ready for application to analysis, but if we need generation we should first reorder the list according to lemmas, and then, given a lemma and a grammar tag, find the corresponding word form.

For example, the reordered fragment of the list above contains:

...board NNS boards
board VBD boarded
board VBG boarding
board VBN boarded
board VBZ boards

boardroom NN boardroom
boardroom NNS boardrooms...

Note that if our morphological generator accepts a word form as the input, we should first apply morphological analysis for generation of the corresponding lemma, and only then call the generator.

Morphological generation is used during correction of the "Noun number", "Subject-Verb Agreement", and "Verb form" error types. It is not used in processing of the "Preposition" and "Article or Determiner" error types.

The other resource that we used is the list of uncountable nouns. The list of 250 most common uncountable nouns is available at www.englishclub.com > Learn English > Vocabulary > Nouns. For example,

...laughter	*measles*	*permission*	*respect*
lava	*meat*	*physics*	*revenge*
leather	*metal*	*poetry*	*rice*
leisure	*methane*	*pollution*	*room*
lightning	*milk*	*poverty*	*rubbish*
linguistics	*money*	*power*	*rum*
literature	*mud*	*pride*	*safety*
litter	*music*	*production*	*salad*
livestock	*nature*	*progress*	*salt*
logic	*news*	*pronunciation*	*sand*
loneliness	*nitrogen*	*psychology*	*satire*
love	*nonsense*	*publicity*	*scenery*
luck	*nurture*	*punctuation*	*seafood*
luggage	*nutrition*	*quality*	*seaside*
machinery	*obedience*	*quantity*	*shame*
magic	*obesity*	*quartz*	*shopping*
mail	*oil*	*racism*	*silence*
management	*oxygen*	*rain*	*sleep*
mankind	*paper*	*recreation*	*smoke*
marble	*passion*	*relaxation*	*smoking...*
mathematics	*pasta*	*reliability*	
mayonnaise	*patience*	*research*	

We used this list for checking the "Noun number" type of errors, when we consider that these nouns should not have the plural form.

Finally, we used the data provided for training by the organizers of the ConLL shared task, i.e., the sentences with syntactic data parsed by

Stanford parser. This data was used to extract syntactic n-grams that correspond to processing of the "Preposition" error type.

We used no other linguistic data. Some of the systems that participated in the task used vast corpora and Internet.

4.2 *Rules of the System*

As we mentioned before, first the "Noun number" error type is processed. We search the plural of the nouns from the list of uncountable nouns. If we find this situation, then we generate the noun in singular and change the verb (agreement) if this noun is a subject.

We made an exception for the noun "time" and do not consider it as uncountable, because its use in the common expressions such as "*many times*" is much more frequent than its use as an uncountable noun as in "*theory of time*" or "*what time is it now*?". Note that word sense disambiguation would be helpful in resolution of the mentioned ambiguities. In addition, the rule which considers the presence of the dependent words like "*many, a lot of, amount of*" could be added.

The next error type is the "Subject verb agreement". We use the very simple rule for verbs in present (with tags VB and VBZ): if its subject is a noun in singular or it is a third person singular pronoun (*he*, *she*, *it*) and the verb is not a modal verb then it should be the verb for third person singular (VBZ). If it is not so, then it is an error and we correct it changing VB to VBZ or vice versa and generating the corresponding verb form.

There are two additional rules for special situations. In case of coordinative construction in the subject we change the grammar number to plural. In case of one or several auxiliary verbs (marked as *aux* or *auxpass*), that auxiliary verb that has the smallest number in the sentence is considered, like, for example, in *have been doing*. This rule exploits fixed word order in English.

The "Verb form" error type includes vast and different types of errors, so we create rules only for some cases of this error type. The rules for verb form correction are as follows: 1) if we have a modal verb, then the depending verb should have a VB tag, 2) if we have an auxiliary verb "have", then the main verb should have a VB tag (perfect tense). We could have created more rules for treatment of situations like "to reforms → to reform", etc. These rules are necessary and would improve the performance, but they cover very small percentage of the data, so for the sake of time we omit further development in this direction.

The error type "Preposition" exploits the previously described concept of syntactic n-grams. In this case, we consider only continuous sn-grams, which are treated as syntactic patterns.

It is well-known that prepositions depend on lexical units that are their heads, for example, see (Eeg-Olofsson, Knutsson 2003), which has been used for, for example, syntactic disambiguation (Galicia-Haro, Gelbukh 2007; Calvo, Gelbukh 2003). In our case, we do not have enough training data, because we use just the available data of the task. So, the performance of our system will be limited to the repetitions of syntactic patterns in the test data.

We conducted several experiments and found out that it is worth considering the dependent word of the preposition as well. Due to very limited training data we are obliged to consider not the word itself, but its POS tag, otherwise our recall would be bad. Note that we consider the neighbors as obtained from the syntactic tree. This method of considering neighbors in syntactic path, instead of taking them directly from the text, corresponds to the previously discussed concept of syntactic n-grams. Here we are talking about mixed syntactic 3-grams. They are mixed because two elements are lexical units (words) and the third element is POS tag. These are continuous sn-grams because bifurcations are not considered.

The fragment of the data that we obtained from the training corpus is presented in Table 2. Total number of the obtained syntactic n-grams is 1,896.

Table 2. Format of the sn-gram data for processing of prepositions.

Wrong prep.	Right prep.	Head word (lemma)	Head word POS tag	Dep. word (lemma)	Dep. word POS tag
in	*for*	*risk*	*NN*	*disorder*	*NN*
from	*of*	*application*	*NN*	*RFID*	*NNP*
into	*	*develop*	*VBZ*	*disease*	*NN*
for	*to*	*advantage*	*NNS*	*human*	*NN*
for	*	*request*	*VBG*	*test*	*NN*

In Table 2, the first column contains the wrong preposition (the error), while the second column has the correct preposition, i.e., the correction. The asterisk corresponds to the absence of the preposition, i.e., the preposition should be deleted. The other columns contain normalized head word with its POS tag and normalized dependent word with its POS tag.

The continuous syntactic 3-grams, which correspond to the rows of the table, are: "*risk in NN → risk for NN*", "*application from NNP → application of NNP*", "*develop into NN → develop NN*", "*advantage for NN → advantage to NN*".

The rule which was implemented in the system is the following: if a relation with preposition is found, then take its head word, POS tag of the dependent word and search in the list of syntactic patterns. If the combination with all three elements is found, then change the preposition to the correct one.

In case of the errors related to "Article or Determiner" type, we only implemented the part related to (1) the choice of the allomorph "*a*" vs "*an*", and (2) the incompatibility of the article "*a*" with nouns in plural. All other rules related to these phenomena take into account discourse information, so they cannot be treated with simple context based rules, even using syntactic information.

5 Scores and Discussion

The results obtained with the evaluation script for our system (Ng, Wu, Wu, Hadiwinoto, Tetreault 2103) for the "SVA/Vform" error types are precision 8.13%, recall 12.42% and F1 measure 9.83%, which was the only error types considered. The results are not very high, though the results of the other systems are not much higher: the average scores are precision 11.82%, recall 20.89%, F1 measure 13.41%, while the best system got precision 17.89%, recall 38.94%, F1 measure 24.51%, but as we mentioned previously, our system uses a rule-based approach with very few additional resources, so it cannot compete with machine learning based approaches that additionally rely on vast lexical resources and the Internet.

Due to its simplicity, low use of additional resources, and very short development time, we position our system as a possible baseline system for the task.

On the other hand, we would like to mention that in some cases the rules should be used as a complementary technique for machine learning methods: don’t guess if you know (Tapanainen, Voutilainen 1994). We consider that the following rules, which are exact, can complement machine learning systems: the rules for the article “a”, the rules for uncountable nouns (in this case, word sense disambiguation would help to determine if the sense in the text is an uncountable noun or has some other use), the subject-verb agreement rules, the rules for correct verb form (here it should be mentioned that these rules cannot cover all errors, but only the most obvious cases).

It is always useful to perform an analysis of the errors committed by a system. Let us analyze the supposed errors committed by our system for the “Noun number” error type.

It performed 18 corrections, 3 of which coincide with the marks in the corpus data. Two of them are clear errors of the system: “*traffic jam*”, where the word “*jam*” is used in a sense other than that of the “substance”, and “*many respects*”, where again the word “*respect*” has a different meaning to that of the uncountable noun. As we mentioned before, WSD techniques should be used to determine the correct sense.

There are 13 cases listed below (in the texts, the word “LIVINGS” is encountered 5 times the word and “QUANTITIES” is encountered two times), that our system marked as errors, because they are uncountable nouns in plural, but they are not marked in the corpus. Let us consider the nouns in capital letters:

peaceful(JJ) LIVINGS(NNS)...,
life(NN) QUALITIES(NNS)...,
Many(JJ) science(NN) FICTIONS(NNS)...,
does(VBZ) not(RB) have(VB) enough(JJ) LANDS(NNS)...,
indicates(VBZ) that(IN) the(DT) FOODS(NNS) the(DT) people(NNS) eat(VBP)...,
problem(NN) of(IN) public(JJ) TRANSPORTATIONS(NNS)...,
healthcare(NN) consume(VBP) large(JJ) QUANTITIES(NNS) of(IN) energy...,
this(DT) society(NN) may(MD) lack(VB) of(IN) LABOURS(NNS)...

Note that the words “equipment” and “usage” in plural were marked as errors in the corpus. In our opinion, it is inconsistent to mark these two as errors, and not to mark the other words from this list as such. While it is true that their use in plural is possible, it is clearly forced and is much less probable. At least, students of English should learn to use these words in singular only. Some of these mistakes (but not all) were corrected by the organizers for the final scoring data.

6 Conclusions

In this paper, we have described a system developed for the CoNLL-2013 shared task: automatic English (as second language, L2) grammar error correction.

The system relies on the rule-based approach. It uses very few additional linguistic data: a morphological analyzer and the list of 250 common uncountable nouns, along with the training data provided by the organizers.

The system uses the syntactic information available in the training data represented as syntactic n-grams, i.e., n-grams extracted by following the paths in dependency trees. These n-grams have certain advantages over traditional n-grams and allow introducing of syntactic information into machine learning.

The system is simple and was developed in a short period of time (2 months, 1 person/months). Since it does not employ any additional resources or sophisticated machine learning methods, it does not achieve high scores, but it could be considered as a baseline system for the task.

On the other hand, it shows what can be obtained using a simple rule-based approach and describes some situations when a rule-based approach can perform better than machine learning method.

ACKNOWLEDGEMENTS This work was done under partial support of the Mexican Government (CONACYT, SNI, COFAA-IPN, SIP-IPN 20120418, 20121823, 20131441, 20131702), CONACYT-DST India ("Answer Validation through Textual Entailment"), Mexico City Government (ICYT DF PICCO10-120), and FP7-PEOPLE-2010-IRSES: "Web Information Quality – Evaluation Initiative (WIQ-EI)" European Commission project 269180. I would like to thank Anubhav Gupta and Martin Tozer for help in implementing programming tasks; Dolors Catala, Angels Catena, and Sandrine Fuentes for help in formulating some rules of the system.

References

1. Bolshakov, I.A., Gelbukh, A.: Computational linguistics: Models, resources, applications. IPN–UNAM–FCE, (2004) 187 pp.
2. Bolshakov, I.A., Gelbukh, A.: Paronyms for Accelerated Correction of Semantic Errors. International Journal on Information Theories and Applications 10 (2003) 11–19
3. Bolshakov, I.A., Galicia-Haro, S.N., Gelbukh, A.: Detection and Correction of Malapropisms in Spanish by means of Internet Search. Lecture Notes in Artificial Intelligence 3658 (2005) 115–122
4. Calvo, H., Gelbukh, A. Improving Prepositional Phrase Attachment Disambiguation Using the Web as Corpus. Lecture Notes in Computer Science 2905 (2003) 604–610
5. Dahlmeier, D., Ng, H.T.: Better evaluation for grammatical error correction. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2012) (2012) 568–572
6. Dahlmeier, D., Ng, H.T., Wu, S.M.: Building a large annotated corpus of learner English: The NUS corpus of learner English (2013)
7. de Marneffe, M.C., MacCartney, B. Manning, C.D.: Generating typed dependency parses from phrase structure parses. In: Proceedings of LREC 2006 (2006)
8. Eeg-Olofsson, J., Knutsson, O.: Automatic grammar checking for second language learners – the use of prepositions. In: Proceedings of NODALIDA’03 (2003)
9. Galicia-Haro, S.N., Gelbukh, A.: Web-Based Model for Disambiguation of Prepositional Phrase Usage. Lecture Notes in Artificial Intelligence 4827 (2007) 922–932
10. Gelbukh, A.: Natural language processing: Perspective of CIC-IPN. In: Proceedings of the International Conference on Advances in Computing, Communications and Informatics, ICACCI 2013, IEEE (2013) 2112–2121
11. Gelbukh, A.: Syntactic disambiguation with weighted extended subcategorization frames. In: Proceedings of PACLING-99, Pacific Association for Computational Linguistics, University of Waterloo, Canada (1999) 244–249
12. Gelbukh, A., Calvo, H. Torres, S.: Transforming a Constituency Treebank into a Dependency Treebank. Procesamiento de Lenguaje Natural 35 (2005) 145-152
13. Gelbukh, A., Kolesnikova, O.: Multiword Expressions in NLP: General Survey and a Special Case of Verb-Noun Constructions. In: Emerging Applications of Natural Language Processing: Concepts and New Research. IGI Global. (2013) 1–21

14. Gelbukh, A., Kolesnikova, O.: Semantic Analysis of Verbal Collocations with Lexical Functions. Studies in Computational Intelligence 414 (2013) 146 pp.
15. Ledeneva, Y., Gelbukh, A., García-Hernández, R.A.: Terms Derived from Frequent Sequences for Extractive Text Summarization. In: Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics, CICLing 2008. Lecture Notes in Computer Science 4919 (2008) 593–604
16. Ng, H.T., Wu, S.M., Wu, Y., Hadiwinoto, C., Tetreault, J.: The CoNLL-2013 Shared Task on Grammatical Error Correction. In: Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task, Bulgary: ACL (2013).
17. Pado, S., Lapata, M.: Dependency-based construction of semantic space models. Computational Linguistics, 33(2) (2007) 161–199.
18. Padró, L., Collado, M., Reese, S., Lloberes, M., Castellón, I.: Freeling 2.1: Five years of open-source language processing tools. In: Proceedings of 7th Language Resources and Evaluation Conference (LREC 2010), ELRA (2010)
19. Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., Chanona-Hernandez, L.: Syntactic dependency-based n-grams as classification features. Lecture Notes in Artificial intelligence 7630 (2012) 1–11.
20. Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., Chanona-Hernandez, L.: Syntactic dependency-based n-grams: More evidence of usefulness in classification. In: Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics, CICLing 2013. Lecture Notes in Computer Science 7816 (2013) 13–24
21. Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., Chanona-Hernandez, L. Syntactic N-grams as machine learning features for natural language processing. Expert Systems with Applications 41(3) (2014) 853–860 (to appear)
22. Sidorov, G.: Non-continuous Syntactic N-grams. Polibits 48 (2013) 67–75 (in Spanish, abstract and examples in English)
23. Tapanainen, P., Voutilainen, A.: Tagging accurately – don't guess if you know. In: Proceedings of ANLP '94 (1994)

GRIGORI SIDOROV
NATURAL LANGUAGE AND TEXT PROCESSING LABORATORY,
CENTRO DE INVESTIGACIÓN EN COMPUTACIÓN,
INSTITUTO POLITÉCNICO NACIONAL (IPN),
MEXICO CITY, MEXICO
WEB: <WWW.CIC.IPN.MX/~SIDOROV>