Improving CoGrOO: the Brazilian Portuguese Grammar Checker

William D. Colen M. Silva[1], **Marcelo Finger**[1]

[1]Instituto de Matemática e Estatística – Universidade de São Paulo (IME/USP)
Rua do Matão, 1010 – Cidade Universitária – São Paulo – SP – Brasil – CEP 05508-090

{colen,finger}@ime.usp.br

Abstract. *This paper highlights the main results obtained in an effort to improve the grammar checker CoGrOO, a hybrid system which initially annotates the text using statistical Natural Language Processing (NLP) techniques, and then apply a rule-based analysis to identify possible grammar errors. The goal was to reduce omissions and false alarms while improving true positives without adding new error rules. The work contributed with a detailed evaluation of low-level NLP modules and the results were compared to state-of-the-art results. The low-level NLP modules are available as open source software, thus improvements on their effectiveness will make them robust, free and ready-to-use alternatives for other systems.*

1. Introduction

Apache OpenOffice is a multi-platform and multilingual office suite, and an open source project[1]. As with its sister project LibreOffice, it is the successor of the OpenOffice.org project, whose first release dates back to 2002. OpenOffice.org did not possess grammar-checking functionality, which made it less competitive compared to other non-open source alternatives. This motivated some NLP researchers to create CoGrOO[2], a Brazilian Portuguese grammar checker, a project initially sponsored by FINEP (a Research and Projects Funding agency). This research on CoGrOO began in 2004, and since its first release in 2006 it has been adopted by important companies like Petrobras - the biggest company in Brazil and the 8th biggest in the world in market value - and Celepar - the Paraná State information technology company, responsible for deploying software for government offices and public schools. CoGrOO accumulated over a hundred thousand downloads from its official website.

CoGrOO is an open source project. It is capable of identifying Portuguese mistakes like pronoun placement, noun agreement, subject-verb agreement, usage of the accent stress marker (`), and other common errors found in Brazilian Portuguese writing.

The CoGrOO grammar checker takes the user's text as input, and outputs a list of possible errors. To accomplish this, it performs a shallow parsing followed by rule-based checking. Initially it analyzes the text using Natural Language Processing (NLP) techniques. The text goes through a pipeline of annotators to identify sentence and token boundaries, to assign a part-of-speech (POS) tag for each word and to find phrase chunks

[1]Apache OpenOffice - Free and Open Productivity suite, `http://openoffice.org`, last accessed on 09/04/2013

[2]CoGrOO official website, `http://cogroo.org`, last accessed on 09/04/2013

and subject-verb relationships. The tagset is based on the Floresta tagset [Afonso 2003]. Then it matches a set of man-made error rules with the resultant structure. An error rule consists of a pattern of words, POS tags, or phrase tags, and some other information, like a description of the error and how an alternative suggestion can be generated.

Although CoGrOO is a successful project considering the number of downloads, which already surpasses 180 thousands[3], its internal components could benefit from several improvements, and that was the work proposed by [Silva 2013], which is surveyed in this work. The main contribution was an analysis of each module of the grammar checker (e.g. the sentence detector, tokenizer, POS tagger, chunker, shallow parser) to determine possible effectiveness bottlenecks and to propose changes to overcome them. To verify the effects of improvements in each module, no new error rules were created to improve coverage; the work focused only on the statistical modules of CoGrOO. Improvements in CoGrOO's modules have further impact in other free software that are based on them, as discussed in Section 6. Expanding the number of error rules was left for future studies.

As reviewed texts are expected to have very few grammar errors, the grammar checker was evaluated against false errors using the reviewed part of the Floresta corpus [Afonso 2003], which is extracted from newspaper articles.

The coverage of grammatical errors is checked with 3 corpora: *a*) a new corpus assembled from real texts submitted by CoGrOO users; *b*) the Metro corpus [Menezes et al. 2006], composed of authentic texts with manually annotated grammar errors; and *c*) the PROBI corpus [Martins 2002], composed of both correct and incorrect sentences, assembled to evaluate ReGra, a proprietary Brazilian Portuguese grammar checker.

The following sections describe the main results; for full details, see [Silva 2013]. Section 2 defines the problem and the objectives. Section 3 presents the architecture of the system, and Section 4 the dictionaries and corpora used during his work. Section 5 describes the development and the experimental results and Section 6 why these improvements in CoGrOO's modules have further impact in other free software. Finally, Section 7 shows future work opportunities and the conclusion.

2. Problem statement and objectives

A written text is subject to errors such as [Kinoshita et al. 2006]:

- Spelling errors: when a word is misspelled, for example as in `sugeito`, while the correct spelling is `sujeito` (subject);
- Grammatical errors: when grammatical rules are not observed, for example in `Nós vai para casa.` (We goes home). This error relates to subject-verb agreement.

Automatic proofreaders are useful to help write documents with fewer mistakes. Currently, CoGrOO only handles grammar error, checking if the text complies with some grammar rules. The task is difficult and complex. There is no established best strategy

[3]CoGrOO downloadscount from Source Forge, `http://sourceforge.net/projects/cogroo/files/stats/timeline?dates=2006-11-13+to+2013-04-09`, last accessed on 09/04/2013

for assembling this kind of application. It requires a number of dictionaries and language models, and it must be computationally efficient.

Grammar checker effectiveness can be evaluated by measuring the number of false alarms versus correctly pointed out error; the former occurs when the grammar checker points out an error where there is none, the latter means the program's goal was achieved. False alarms are known as *false positives* and the fewer the better. Correctly pointed out errors are known as *true positives*, and the higher the better.

The goal was to increase the number of true positives and decrease the number of false positives, without creating any new error rule, which was left for future studies.

3. Proposed system architecture

The Grammar Checker CoGrOO is a hybrid NLP system, in the sense that it employs statistical analysis to perform text segmentation and categorization, and uses hand-written rules to find error patterns. The system is composed of the following modules organized as a pipeline:

1. Sentence Boundary Detector: receives a text as input and divides it into sentences;
2. Tokenizer: receives a sentence and divides it into words and punctuation marks;
3. Name Finder: receives the sentence tokens and identifies the potential proper nouns, such as person, places and organization names, and merges them as one token;
4. Contraction Finder: receives the sentence tokens and identifies Portuguese contractions, such as `na` which is the contraction of `em + a`, and expands them;
5. Part-of-Speech Tagger: receives a sentence and assigns the most probable morphological tag to its lexical items, according to their context;
6. Featurizer: receives the tokens and the morphological tags associated to them, and outputs features such as gender, number, tense and person;
7. Chunker: receives a tagged text and finds its noun phrases, verbal phrases, prepositional phrases and adverbial phrases;
8. Chunk-Head Finder: receives a tagged sentence with the associated chunks and searches for the head of each chunk;
9. Shallow Parser: receives a tagged sentence with the associated chunks and searches for structures such as the predicate, subject and object of the sentence;
10. Grammar Error Detector: this module looks for grammar errors in the input sentence. It is activated after all the previous sentence analysis steps have been done.

This architecture, which is shown in Figure 1, differs from previous versions of the grammar checker. A Contraction Finder was already available, but it was unable to use contextual information to decide whether ambiguous tokens are contractions. For example, the token `consigo` can be the contraction of the preposition `com` and the pronoun `sigo`, or the present form of the verb `conseguir`, but previous version would always split it as if it were a contraction.

Another change is the introduction of the dedicated Featurizer module. In the previous versions, the featurizaton task was performed by the POS Tagger, which suffered from data sparseness due to the augmented tagset. Similarly with the Chunk-Head Finder, whose functionality was previously handled by the Chunker. A deeper justification for these changes can be found in [Silva 2013].

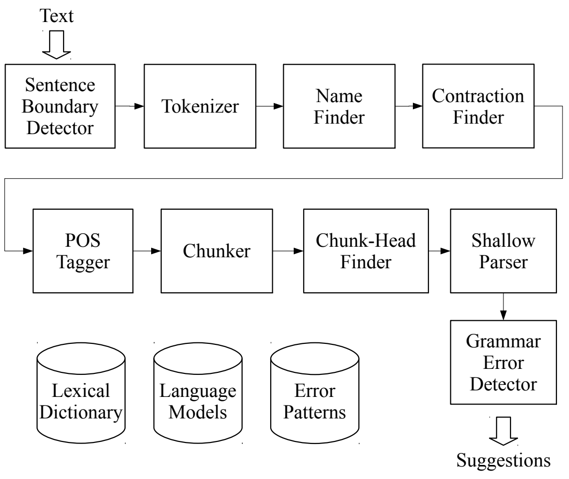

Figure 1. Grammar checker processing pipeline

A chunk-head is used by the algorithm of the Grammar Error Detector to assign a gender and a number to noun phrases. The module was trained using a binary tagset, which simply denotes whether a token is a chunk-head or not. The training data could be easily derived from the Portuguese corpora detailed in the next section.

All modules but the Grammar Error Detector were built using statistical machine learning techniques; more specifically, the Maximum Entropy [Ratnaparkhi 1996] and Perceptron [Collins 2002] frameworks. For most of the tasks, Perceptron proved consistently more effective compared to Maxent. CoGrOO relies on implementations from Apache OpenNLP framework.

The Grammar Error Detector is responsible for detecting possible errors in the input sentence through the use of hand-written error rules. An error rule is just a pattern that is searched in the input sentence. The rule patterns are compiled into a Finite State Machine and applied to the text. Aiming to observe how the other modules affect the effectiveness of the Grammar Error Detector, this module is not changed in this work.

4. Linguistic resources

A number of linguistic resources were used to develop the grammar checker: corpora, which are used both for training the machine learning modules, as well as to evaluate the grammar checker; and lexical dictionaries, which are used to support the POS Tagger and Featurizer tasks.

4.1. Corpora

Two different types of corpora were used during the development of this work: *a*) corpora annotated with morphology and syntax; and *b*) corpora annotated with grammar errors. The first type was used to train the statistical machine learning modules of the grammar checker, while the latter to evaluate the grammar checker effectiveness.

To train the statistical machine learning modules two corpora were effectively used. The first was Floresta Virgem (Virgin Forest), which is a set of trees automatically created from the Constraint Grammar (CG) output of the PALAVRAS parser[4]. It is a 96,000-sentence corpus, with both Brazilian and European variants of Portuguese, but only the Brazilian variant was used. The Brazilian portion is called the CETENFolha, and is a subset of the NILC Corpus. Its text material comes from the Folha de São Paulo newspaper. This corpus is useful for training and test modules when quantity is better than quality.

The second was Bosque (Grove) which is a subset of Floresta Virgem, with fully revised annotations of 186,000 words and 9,368 sentences, again with both Brazilian and European variants of Portuguese, but only the Brazilian variant were used. It is available in flat CG format, or as trees in Árvores Deitadas format (Lying Trees) (AD) [Afonso 2006] according to the Floresta Symbolset[5]. From AD format, it is easier to extract structural information, like chunks and clauses. This corpus is useful to train and test modules when quality is better than quantity.

To evaluate the grammar checker four corpora were used. The first is the previously mentioned Bosque corpus, which being a professionally revised text is supposed to have very few grammar errors. This corpus is used to measure the number of false positives.

PROBI [Martins 2002] is the second used corpus. Developed to evaluate the ReGra grammar checker, it is composed of 11,625 sentences supposedly verifiable in the written record of Brazilian Portuguese users. It provides: *a*) a set of correct and legitimate sentences; *b*) a set of sentences which, although belonging to dialectal varieties of Brazilian Portuguese and practised by a significant subset of native speakers, are not considered standard Portuguese nor recommended or tolerated by dictionaries and grammars; *c*) a set of sentences with typos or editing mistakes that do not belong to any variety of Portuguese. 2,616 of its sentences have mistakes, which are categorized.

The third is the Metrô Corpus, which was assembled to evaluate an old version of CoGrOO grammar checker [Menezes et al. 2006]. The texts were extracted from the São Paulo subway company's institutional website [Metro 2011], in December 2006. From its 781 sentences, 53 were manually annotated as having grammar errors.

Finally, the CoGrOO Community Corpus. CoGrOO Community is a web application that provides collaborative tools for grammar-checker users. Users of CoGrOO were invited to join the Community portal and submit sentences that caused CoGrOO to fail, indicating whether it is a false negative or false positive. In each case the user provides relevant classification details. The corpus has 457 sentences from which 188 were manually annotated as having grammar errors.

4.2. Dictionary

The lexical dictionaries are used to support the POS tagging task, for example, by restricting which tags should be assigned to tokens. The entries in a lexical dictionary include word class and inflection information.

[4]The Constraint Grammar category set of Palavras `http://beta.visl.sdu.dk/visl/pt/info/portsymbol.html`, last accessed on 09/04/2013

[5]Grammatical categories (tags) used in the Floresta project `http://beta.visl.sdu.dk/visl/pt/info/symbolset-floresta.html`, last accessed on 09/04/2013

The CoGrOO lexical dictionary derives from JSpell, which is a morphological analysis tool created by the Natura Project [Almeida 2011]. Jspell.br is an initiative of the CoGrOO team and its contributors which objective is to translate the JSpell dictionary to the Brazilian variant of Portuguese. The work is currently under development, but the dictionaries are already available and can be used by CoGrOO modules. Both dictionary and sources are available online[6].

5. Experiments and development

The objective of the experimental phase was to spot bottlenecks of the grammar checker processing pipe, especially those causing false positives. The experiments were divided into three phases:

Initial evaluation In this phase the current version of CoGrOO, version 3.1.2, is tested for true positives and false positives using the evaluation corpora.

NLP Development and Evaluation During this phase, each of the CoGrOO proposed modules was individually developed and tested. The development consists of the creation of a new implementation module, followed by an evaluation of each different implementation using a 10-fold cross-validation. Finally, each implementation is evaluated by measuring its impact in the grammar checker effectiveness. After this process, the best module is added to the new grammar checker baseline, which is used in the development of the following modules.

Final evaluation The final evaluation measured how the new modules impacted the grammar checker effectiveness and compares the results with the initial evaluation.

A detailed discussion of all experiments can be found in [Silva 2013], but due to space restrictions, only the final results are shown here.

5.1. Low-level NLP modules

Table 1 highlights the evaluation results of the low-level NLP components, and Table 2 the configuration which obtained the best result. It is important to notice that for this work not only the best F_1 or accuracy score was important, but also the impact of the configuration in the grammar checker effectiveness.

According to Table 2, Perceptron models performed better than Maximum Entropy (Maxent) for all modules but for Chunk Head Finder. The modules that benefited from the size of Floresta Virgem CETENFolha (VCF) were Sentence Detector, Tokenizer and Contraction Finder, while the others were more effective while trained with Bosque CETENFolha (CF).

The Apache OpenNLP defaults were sufficient to train efficient models for Name Finder and Chunker, but others required some additional tuning. This was the case of the Sentence Detector, which F_1 raised from 98.928% to 99.237% with the additional contextual predicates (ACP) designed for Portuguese and the abbreviation dictionary (ABB).

[6]JSpell.br website, `http://github.com/cogroo/jspell.br`, last accessed on 09/04/2013

Module	Accuracy	Precision	Recall	F_1
Sentence Detector		99.215	99.259	99.237
Tokenizer		99.949	99.933	99.941
Name Finder		89.019	86.002	87.484
Contraction Finder		99.948	99.945	99.947
POS Tagger	96.291			
Featurizer	96.743			
Chunker		96.341	96.431	96.386
Chunk Head Finder	99.777			
Shallow Parser		83.818	82.493	83.150

Table 1. Final 10-fold cross-validation score for each of the CoGrOO models.

Tokenizer also benefited by the abbreviation dictionary and by the alphanumeric optimization (AO), which skips tokens composed only by alphanumeric characters, and the F_1 raised from 99.921% to 99.941%. To the default POS Tagger were added: the JSpell.br dictionary (DIC), as well as a dictionary based on the training data (MDIC); additional context from the Name and Contraction Finder modules (AC), which allows using decisions taken by that modules as features; and a custom context generator (CCG) to boost the handling of the tokens `a` and `que`. This improved the accuracy from 94.672% to 96.291%. All these values refer to 10-fold cross-validation experiments.

Contraction Finder, Featurizer, Chunker Head Finder and Shallow Parser were implemented from scratch because there are no OpenNLP modules for such tasks.

Module	Algorithm	Cutoff	Corpus	Options
Sentence Detector	Perceptron	0	VCF	ACP, ABB
Tokenizer	Perceptron	0	VCF	ABB, AO
Name Finder	Perceptron	0	CF	
Contraction Finder	Perceptron	8	VCF	
POS Tagger	Perceptron	0	CF	DIC, MDIC, CCG, AC
Featurizer	Perceptron	0	CF	DIC, WHNCS
Chunker	Perceptron	0	CF	
Chunk Head Finder	Maxent	8	CF	
Shallow Parser	Perceptron	0	CF	

Table 2. The configurations which generated the best models.

Table 3 shows how the changes in the low-level modules affected the grammar checker effectiveness. The changes affected especially the precision. For PROBI corpus the precision raised from 23.21% to 60.03%, and for Metro corpus from 19.28% to 44.12%. For Bosque corpus, which have been proof-read, the number of false positives decreased from 245 to 69 in 9,368 sentences. This improvement is significant to the user experience because the less false positives, the less the user is interrupted by the grammar checker and the higher is the user's trust in it.

Recall values did not improve significantly. To improve recall it would be required adding new error rules, which was left for future studies.

Category	Version	Precision	Recall	F_1	Target	TP	FP
PROBI	3.1.2	23.21%	13.65%	17.19%	2616	357	1181
	4.0.0	60.03%	14.41%	23.24%		377	251
Metrô	3.1.2	19.28%	30.19%	23.53%	53	16	67
	4.0.0	44.12%	28.30%	34.48%		15	19
Comunidade	3.1.2	2.47%	3.85%	3.01%	104	4	158
	4.0.0	44.12%	28.30%	34.48%		24	61
Bosque	3.1.2						245
	4.0.0						69

Table 3. Effectiveness results for each corpus. CoGrOO Version 3.1.2 was original without improvements, and 4.0.0 is the improved version with modifications proposed in [Silva 2013].

6. Impact

Improving the grammar checker not only benefited its community of users, which now has access to a more accurate software, but also to a number of other projects, because its internal modules are in use by a range of projects.

For example, OnAIR (Ontology Aided Information Retrieval) is a system developed for searching within digital video databases. CoGrOO POS Tagger and Chunker is used as part of the computational tools to analyse audio transcription linguistic information [Torres 2012], and also as part of a pipeline to analyse natural language queries [Luz 2012].

In health informatics, a study applies CoGrOO annotators to the task of Information Retrieval on Clinical Data [Oleynik 2012], another to perform pattern identification in discharge summaries [Souza 2012], and finally to a study which classifies whether there is continuity of care in hospital discharge summaries [Oliveira et al. 2012].

In linguistic research, CoGrOO is used to annotate texts with morphology and syntactic information, and later allow researchers to use these annotations in semiotic research [Matte et al. 2012].

7. Future Work and Conclusion

The work of [Silva 2013] left a large number of future work opportunities. For example, the work did not cover the introduction of new error rules to improve coverage. Adding a machine learning component trained to identify grammar errors is a challenge that would also improve coverage. Finally, an individual and deep study of key modules, like the shallow parser, or the introduction of new modules, like a clause identifier, would boost the analysis power and consequently the grammar checker effectiveness.

The work presented was broad in the sense that it covers many of the basic Natural Language Processing tasks, but it is also shallow because it does not deepen any of these tasks, even though frequently the effectiveness of the modules achieved values close to the state-of-the-art. The main purpose of his work was to improve the Portuguese grammar checker by tuning its language processing modules, which as shown, was very effective.

References

Afonso, S. (2003). A floresta sintá(c)tica como recurso. *Documentaçao disponıvel na Linguateca*.

Afonso, S. (2006). Árvores deitadas: Descrição do formato e descrição das opções de análise na floresta sintá(c)tica. Technical report, Floresta Sintática.

Almeida, J. (2011). Natura project, natural language processing group. `http://natura.di.uminho.pt/wiki/doku.php?id=projectonatura`.

Collins, M. (2002). Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 1–8. Association for Computational Linguistics.

Kinoshita, J., Salvador, L. N., and de Menezes, C. E. D. (2006). Cogroo: a brazilian-portuguese grammar checker based on the cetenfolha. In *Proceedings LREC 2006*.

Luz, F. F. (2012). Consulta à ontologias em língua portuguesa através do português controlado. Master's thesis, Computer Science Department, University of São Paulo.

Martins, R. (2002). Probi: um corpus de teste para o revisor gramatical regra. `http://www.nilc.icmc.usp.br/nilc/download/NILC-TR-02-10.zip`.

Matte, A. C. F., Ribeiro, R. T., de Moura Silva, W. D. C., and Canalli, H. L. (2012). Dadossemiotica: coleta e processamento de análises semióticas de texto escrito. In *WSL Workshop Internacional de Software Livre*.

Menezes, C. E. D., Gusukuma, F. W., and Uliano, S. (2006). Uma análise do cogroo, um corretor gramatical acoplável ao openoffice. `http://www.pcs.usp.br/~cogroo/papers/analise-cogroo-corpus-metro.html`.

Metro (2011). `http://www.metro.sp.gov.br`.

Oleynik, M. (2012). Extração de informações de narrativas clínicas. Master's thesis, Computer Science Department, University of São Paulo.

Oliveira, L. E. S., Moro, C. M. C., Souza, A. C., Nohama, P., and Cancian, P. S. (2012). Identificação de continuidade de cuidado em sumários de alta hospitalar. In *XIII Congresso Brasileiro em Informática em Saúde*.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. In *Proceedings of the conference on empirical methods in natural language processing*, volume 1, pages 133–142.

Silva, W. D. C. M. (2013). Refining the CoGrOO grammar checker. Master's thesis, Computer Science Department, University of São Paulo.

Souza, A. C. (2012). Identificação do Conteúdo Padronizado do Sumário de Alta. Master's thesis, PUCPR.

Torres, C. E. A. (2012). Uso de informação linguística e análise de conceitos formais no aprendizado de ontologias. Master's thesis, Computer Science Department, University of São Paulo.