Learning How to Conjugate the Romanian Verb. Rules for Regular and Partially Irregular Verbs

Liviu P. Dinu
Faculty of Mathematics
and Computer Science
University of Bucharest
ldinu@fmi.unibuc.ro

Vlad Niculae
Faculty of Mathematics
and Computer Science
University of Bucharest
vlad@vene.ro

Octavia-Maria Şulea
Faculty of Foreign Languages
and Literatures
Faculty of Mathematics
and Computer Science
University of Bucharest
mary.octavia@gmail.com

Abstract

In this paper we extend our work described in (Dinu et al., 2011) by adding more conjugational rules to the labelling system introduced there, in an attempt to capture the entire dataset of Romanian verbs extracted from (Barbu, 2007), and we employ machine learning techniques to predict a verb's correct label (which says what conjugational pattern it follows) when only the infinitive form is given.

1 Introduction

Using only a restricted group of verbs, in (Dinu et al., 2011) we validated the hypothesis that patterns can be identified in the conjugation of the Romanian (partially irregular) verb and that these patterns can be learnt automatically so that, given the infinitive of a verb, its correct conjugation for the indicative present tense can be produced. In this paper, we extend our investigation to the whole dataset described in (Barbu, 2008) and attempt to capture, beside the general ending patterns during conjugation, as much of the phonological alternations occuring in the stem of verbs (apophony) from the dataset as we can.

Traditionally, Romanian has received a Latin-inspired classification of verbs into 4 (or sometimes 5) conjugational classes based on the ending of their infinitival form alone (Costanzo, 2011). However, this infinitive-based classification has proved itself inadequate due to its inability to account for the behavior of partially irregular verbs (whose stems have a smaller number of allomorphs than the completely irregular) during their conjugation.

There have been, thus, numerous attempts throughout the history of Romanian Linguistics to give other conjugational classifications based on the way the verb actually conjugates. Lombard (1955), looking at a corpus of 667 verbs, combined the traditional 4 classes with the way in which the biggest two subgroups conjugate (one using the suffix ”ez”, the other ”esc”) and arrived at 6 classes. Ciompec (Ciompec et. al., 1985 in Costanzo, 2011) proposed 10 conjugational classes, while Felix (1964) proposed 12, both of them looking at the inflection of the verbs and number of allomorphs of the stem. Romalo (1968, p. 5-203) produced a list of 38 verb types, which she eventually reduced to 10.

For the purpose of machine translation, Moisil (1960) proposed 5 regrouped classes of verbs, with numerous subgroups, and introduced the method of letters with variable values, while Papastergiou et al. (2007) have recently developed a classification from a (second) language acquisition point of view, dividing the 1st and 4th traditional classes into 3 and respectively 5 subclasses, each with a different conjugational pattern, and offering rules for alternations in the stem.

Of the more extensive classifications, Barbu (2007) distinguished 41 conjugational classes for all tenses and 30 for the indicative present alone, covering a whole corpus of more that 7000 contemporary Romanian verbs, a corpus which was also used in the present paper. However, her classes were developed on the basis of the suffixes each verb receives during conjugation, and the classification system did not take into account the alternations occuring in the stem of irregular and partially irregular verbs. The system of rules presented below took into account both the endings pattern and the type of stem alternation for each verb.

In what follows we describe our method for labeling the dataset and finding a model able to predict the labels.

2 Approach

The problem which we are aiming to solve is to determine how to conjugate a verb, given its infinitive form. The traditional infinitive-based classification taught in school does not take one all the way to solving this problem. Many conjugational patterns exist within each of these four classes.

2.1 Labeling the dataset

Following our own observations, the alternations identified in (Papastergiou et al., 2007) and the classes of suffix patterns given in (Barbu, 2007), we developed a number of conjugational rules which were narrowed down to the 30 most productive in relation to the dataset. Each of these 30 rules (or patterns) contains 6 regular expressions through which the rule models how a (different) type of Romanian verb conjugates in the indicative present. They each consist of 6 regular expressions because there are three persons (first, second, and third) times two numbers (singular and plural).

Rule 10, for example, models, as stated in the list that follows, how verbs of the type ”a cânta” (to sing) conjugate in the indicative present, by having the first regular expression model the first person singular form ”(eu) cânt” (in regular expression format: ˆ(.+)$), the second, model the second person singular form ”(tu) cânţi” (ˆ(.+)ţi$), the third, model the third person singular form ”(ei) cântă” (ˆ(.+)ă$), and so forth. Thus, rule 10 catches the alternation t→ţ for the 2nd person singular, while modelling a particular type of verb class with a particular set of suffixes. Note that the dot accepts any letter in the Romanian alphabet and that, for each of the six forms, the value of the capturing groups (those between brackets) remains constant, in this case *cân*. These groups correspond to all parts of the stem that remain unchanged and ensure that, given the infinitive and the regular expressions, one can work backwards and produce the correct conjugation.

For a clearer understanding of one such rule, Table 1 shows an example of how the verb ”a tresălta” is modeled by rule 14.

Below, we list all the rules used, with the stem alternations they capture and an example of a verb

Person	Regexp	Example
1st singular	ˆ(.+)a(.+)t$	tresalt
2nd singular	ˆ(.+)a(.+)ţi$	tresalţi
3rd singular	ˆ(.+)a(.+)tă$	tresaltă
1st plural	ˆ(.+)ă(.+)tăm$	tresăltăm
2nd plural	ˆ(.+)ă(.+)taţi$	tresăltaţi
3rd plural	ˆ(.+)a(.+)tă$	tresaltă

Table 1: Rule 14 modelling ”a tresălta”

that they model. Note that, when we say (no) alternation, we mean (no) alternation in the stem. So the difference between rules 1, 20, 22, and the sort lies in the suffix that is added to the stem for each verb form. They may share some suffixes, but not all and/or not for the same person and number.

1. no alternation; ”a spera” (to hope);
2. alternation: ă→e for the 2nd person singular; ”a număra” (to count);
3. no alternation; ”a intra” (to enter), stem ends in ”tr”, ”pl”, ”bl” or ”fl” which determines the addition of ”u” at the end of the 1st person singular form;
4. alternation: it lacks t→ţ for the 2nd person singular, which otherwise normally occurs; ”a mişca” (to move), stem ends in ”şca”;
5. no alternation; ”a tăia” (to cut), ends in ”ia” and has a vowel before;
6. no alternation; ”a speria” (to scare), ends in ”ia” and has a consonant before;
7. no alternation; ”a dansa” (to dance), conjugated with the suffix ”ez”;
8. no alternation; ”a copia” (to copy), conjugated with a modified ”ez” due to the stem ending in ”ia”;
9. altenation c→ch(e) or g→gh(e); ”a parca” (to park), conjugated with ”ez”, ending in ”ca” or ”ga”;
10. alternation: t→ţ for the 2nd person singular; ”a cânta” (to sing);
11. alternation: s→ş which replaces the usual t→ţ for the 2nd person singular; ”a exista” (to exist);

12. alternation: a→ea for the 3rd person singular and plural, t→ţ for the 2nd person singular; ”a deştepta” (to awake/arouse);
13. alternation: e→ea for the 3rd person singular and plural, t→ţ for the 2nd person singular; ”a deşerta” (to empty);
14. alternation: ă→a for all the forms except the 1st and 2nd person plural; ”a tresălta” (to start, to take fright);
15. alternation: ă→a in the 3rd person singular and plural, ă→e in the 2nd person singular; ”a desfăta” (to delight);
16. alternation: ă→a for all the forms except for the 1st and 2nd person plural; ”a părea” (to seem);
17. alternation: d→z for the 2nd person singular due to palatalization, along with ă→e; ”a vedea” (to see), stem ends in ”d”;
18. alternation: ă→a for all forms except the 1st and 2nd person plural, d→z for the 2nd person singular due to palatalization; ”a cădea” (to fall);
19. no alternation; ”a veghea” (to watch over), conjugates with another type of ”ez” ending pattern;
20. no alternations; ”a merge” (to walk), receives the typical ending pattern for the third conjugational class;
21. alternation: t→ţ for the 2nd person singular; ”a promite” (to promise);
22. no alternation; ”a scrie” (to write);
23. alternations: şt→sc for the 1st person singular and 3rd person plural; ”a naşte” (to give birth), ends in ”şte”;
24. alternation: ”n” is deleted from the stem in the 2nd person singular; ”a pune” (to put), ends in ”ne”;
25. alternation: d→z in the 2nd person singular due to palatalization; ”a crede” (to believe), stem ends in ”d”;
26. no alternation; ”a sui” (to climb), ends in ”ui”, ”ăi”, or ”âi”;
27. no alternation; ”a citi” (to read), conjugates with the suffix ”esc” ;
28. this type preserves the ”i” from the infinitive; ”a locui” (to reside), ends in ”ăi”, ”oi”, or ui” and conjugates with ”esc”;
29. alternation: o→oa in the 3rd person singular and plural; end in ”î”, ”a omorî” (to kill);
30. no alternation; ”a hotărî” (to decide), ends in ”î” and conjugates with ”ăsc”, a variant of ”esc”

2.2 Classifiers and features

Each infinitive in the dataset received a label corresponding to the first rule that correctly produces a conjugation for it. This was implemented in order to reduce the ambiguity of the data, which was due to some verbs having alternate conjugation patterns. The unlabeled verbs were thrown out, while the labeled ones were used to train and evaluate a classifier.

The context sensitive nature of the alternations leads to the idea that n-gram character windows are useful. In the preprocessing step, the list of infinitives is transformed to a sparse matrix whose lines correspond to samples, and whose features are the occurence or the frequency of a specific n-gram. This feature extraction step has three free parameters: the maximum n-gram length, the optional binarization of the features (taking only binary occurences instead of counts), and the optional appending of a terminator character. The terminator character allows the classifier to identify and assign a different weight to the n-grams that overlap with the suffix of the string.

For example, consider the English infinitive *to walk*. We will assume the following illustrative values for the parameters: n-gram size of 3 and appending the terminator character. Firstly, a terminator is appended to the end, yielding the string *walk$*. Subsequently, the string is broken into 1, 2 and 3-grams: *w*, *a*, *l*, *k*, *$*, *wa*, *al*, *lk*, *k$*, *wal*, *alk*, *lk$*. Next, this list is turned into a vector using a standard process. We have first built a dictionary of all the n-grams from the whole dataset. These, in order, encode the features. The verb *(to) walk* is therefore encoded as a row vector with ones in the columns corresponding to the features *w*, *a*, etc. and zeros in the rest. In this particular case, there is no difference between binary and count

rule no.	verbs	rule no.	verbs
1	547	16	13
2	8	17	6
3	18	18	4
4	5	19	14
5	8	20	124
6	16	21	25
7	3330	22	15
8	273	23	7
9	89	24	41
10	4	25	51
11	5	26	185
12	4	27	1554
13	106	28	486
14	13	29	5
15	5	30	27

Table 2: Number of verbs captured by each of our rules

features because all of the n-grams of this short verb occur only once. But for a verb such as *(to) tantalize*, the feature corresponding to the 2-gram *ta* would get a value of 2 in a count reprezentation, but only a value of 1 in a binary one.

The system was put together using the scikit-learn machine learning library for Python (Pedregosa et al., 2011), which provides a fast, scalable implementation of linear support vector machines based on *liblinear* (Fan et al., 2008), along with n-gram extraction and grid search functionality.

3 Results

Tabel 2 shows how well the rules fitted the dataset. Out of 7,295 verbs in the dataset, 349 were uncaptured by our rules. As expected, the rule capturing the most verbs (3,330) is the one modelling those from the 1st conjugational class (whose infinitives end in ”a”) which conjugate with the ”ez” suffix and are regular, namely rule 7, created for verbs like ”a dansa”. The second largest class, also as expected, is the one belonging to verbs from the 4th conjugational group (whose infinitives end in ”i”), which are regular, meaning no alternation in the stem, and conjugate with the ”esc” suffix. This class is modeled by rule number 27.

The support vector classifier was evaluated using a 10-fold cross-validation. The multi-class problem is treated using the one-versus-all scheme. The parameters chosen by grid search are a maximum n-gram length of 5, with appended terminator and with non-binarized (count) features. The estimated correct classification rate is 90.64%, with a weighted averaged precision of 80.90%, recall of 90.64% and F_1 score of 89.89%. Appending the artificial terminator character '$' consistently improves accuracy by around 0.7%. Because each word was represented as a bag of character n-grams instead of a continuous string, and because, by its nature, a SVM yields sparse solutions, combined with the evaluation using cross-validation, we can safely say that the model does not overfit and indeed learns useful decision boundaries.

4 Conclusions and Future Works

Our results show that the labelling system based on the verb conjugation model we developed can be learned with reasonable accuracy. In the future, we plan to develop a multiple tiered labelling system that will allow for general alternations, such as the ones occuring as a result of palatalization, to be defined only once for all verbs that have them, taking cues from the idea of letters with multiple values. This, we feel, will highly improve the acuracy of the classifier.

5 Acknowledgements

The authors would like to thank the anonymous reviewers for their helpful comments. All authors contributed equally to this work. The research of Liviu P. Dinu was supported by the CNCS, IDEI - PCE project 311/2011, ”The Structure and Interpretation of the Romanian Nominal Phrase in Discourse Representation Theory: the Determiners.”

References

Ana-Maria Barbu. *Conjugarea verbelor româneşti. Dicţionar: 7500 de verbe româneşti grupate pe clase de conjugare*. Bucharest: Coresi, 2007. 4th edition, revised. (In Romanian.) (263 pp.).

Ana-Maria Barbu. Romanian lexical databases: Inflected and syllabic forms dictionaries. In *Sixth International Language Resources and Evaluation (LREC'08)*, 2008.

Angelo Roth Costanzo. *Romance Conjugational Classes: Learning from the Peripheries*. PhD thesis, Ohio State University, 2011.

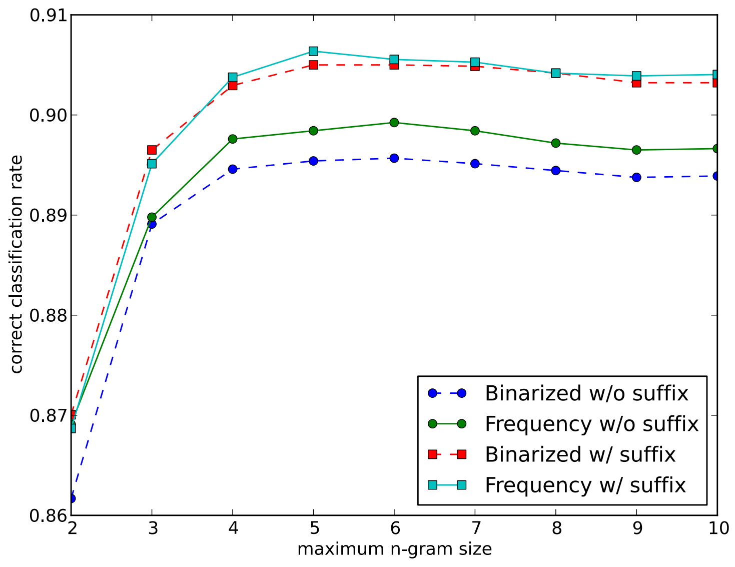

Figure 1: 10-fold cross validation scores for various combination of parameters. Only the values corresponding to the best C regularization parameters are shown.

Liviu P. Dinu, Emil Ionescu, Vlad Niculae, and Octavia-Maria Şulea. Can alternations be learned? a machine learning approach to verb alternations. In *Recent Advances in Natural Language Processing 2011*, September 2011.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, June 2008. ISSN 1532-4435.

Jiři Felix. *Classification des verbes roumains*, volume VII. Philosophica Pragensia, 1964.

Alf Lombard. *Le verbe roumain. Etude morphologique*, volume 1. Lund, C. W. K. Gleerup, 1955.

Grigore C. Moisil. Probleme puse de traducerea automată. conjugarea verbelor în limba română. *Studii si cercetări lingvistice*, XI(1): 7–29, 1960.

I. Papastergiou, N. Papastergiou, and L. Mandeki. Verbul românesc - reguli pentru înlesnirea însuşirii indicativului prezent. In *Romanian National Symposium "Directions in Romanian Philological Research", 7th Edition*, May 2007.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, Oct 2011.

Valeria Guţu Romalo. *Morfologie Structurală a limbii române*. Editura Academiei Republicii Socialiste România, 1968.