Rule-based Surface Realization of Romanian Weak Pronouns

Ciprian-Virgil Gerstenberger
UiT The Arctic University of Norway
ciprian.gerstenberger@gmail.com

Abstract

Due to its reliance on context and intricate grammatical rules, the Romanian weak pronoun system presents a challenge not only for language learners – both native and non-native speakers – but also for linguistic description and computational processing.

The present work addresses the challenges of Romanian weak pronouns from a computational processing perspective. Accordingly, it has three main goals: (1) to present the implementation of a rule-based model for generating contextually accurate surface forms of Romanian weak pronouns, (2) to describe the compilation of a database of relevant inputs for testing surface realization, and (3) to test the effectiveness of the model.

This serves as a proof of concept, demonstrating both the transparency and the effectiveness of the model when based on an appropriate linguistic description.

1 Introduction

Romanian weak pronouns (henceforth, RWPs), commonly referred to as clitics, exhibit a variety of surface forms governed by intricate, contextually dependent morphophonological rules. As a result, they pose challenges not only for linguistic description but also for computational modeling and natural language processing (henceforth, NLP).

Despite extensive research on Romanian clitics within various theoretical frameworks – including Generative Grammar, as explored by Dobrovie-Sorin (1999), Somesfalean (2007), and Săvescu Ciucivara (2009); Lexical Functional Grammar (LFG), as proposed by Barbu and Toivonen (2018); Head-Driven Phrase Structure Grammar (HPSG), as discussed in Monachesi (2001) and Monachesi (2005); various Optimality Theory (OT) models, advanced by Popescu (2000), Sasaki and Căluianu (2000), Legendre (2001), Popescu (2003), and Cherecheș (2014); and Dynamic Syntax, as promoted by Klein (2007) – there remains a need for practical, computational tools that can accurately generate and predict their usage in real-world contexts.

The current study addresses this gap by focusing on the creation of a rule-based model designed to handle the surface realization of RWPs, tested using a specialized input database. This, in turn, contributes to the broader goal of enhancing computational linguistic applications for Romanian.

2 Romanian Weak Pronouns

As with other Romance languages, Romanian employs a pronominal system that includes both strong pronouns and weak (or clitic) pronouns. Romanian has a complex clitic system with fully marked case, number, and gender distinctions, comprising both personal and reflexive RWPs. Moreover, there are some Romanian weak verb (henceforth, RWV) forms of the verb *to be* that behave similarly to RWPs. RWPs can appear in both syllabic and asyllabic forms. Their syllabic surface form, as well as the requirement for asyllabicity (obligatory sandhi) or the possibility of asyllabicity (optional sandhi), is determined by their contexts of occurrence. Sandhi refers to phonological adjustments that occur at morpheme or word boundaries, influenced by various factors. The asyllabic (and thus reduced) RWP

Proceedings of the 9th Workshop on Constraint Grammar and Finite State NLP, pages 9–18
March 5, 2025 ©2025 University of Tartu Library

forms are referred to as sandhi forms.

RWPs occur in a fixed order of up to three elements [RWP_*dat_ethicus* < RWP_*dat* < RWP_*acc*] in clitic clusters (or clitic sequences), alongside other clitics such as negation, auxiliary verbs, or adverbial particles (ex. 1). The cluster can occur both in preverbal position (e.g., in declarative sentences, ex. 2) and postverbal position (e.g., in imperative sentences, ex. 3), with the exception of negation and adverbial particles, which can occur only preverbally.

(1) Nu ni le- ai
not cl_1.pl.dat cl_3.pl.acc.f have_2.sg.pres
mai dat.
more given
«You didn't give them to us anymore.»

(2) Mi le dai
cl_1.sg.dat cl_3.pl.acc.f give_2.sg.pres
acum.
now
«You give them to me now.»

(3) Dă -mi -le acum!
give_2.sg.imp cl_1.sg.dat cl_3.pl.acc.f now
«Give them to me now!»

(4) Le dai mere.
cl_3.pl.dat give_2.sg.pres apple_pl.acc
«You give apples to them.»

Romanian exhibits various types of ambiguity that complicate the interpretation of the RWP data, such as case syncretism between accusative (*le* in ex. 2) and dative (*le* in ex. 4) plural, part-of-speech homonymy (between the dative-reflexive RWP *și* and the conjunction *și* «and» in ex. 5), phoneme-grapheme ambiguity (syllabic form [mi] in ex. 6 vs. asyllabic form with a glide [mi̯] in ex. 7), as well as hyphen ambiguity (marking asyllabicity in *mi-o* in ex. 7 vs. marking postverbality in *dă-mi-le* in ex. 3).

(5) Și le cumpără
cl_3.sg.dat.refl cl_3.pl.acc.f buy_3.sg.pres
și și le
and_conj cl_3.sg.dat.refl cl_3.pl.acc.f
revinde.
resell_3.sg.pres
«He/she buys them for him-/herself and resells them for him-/herself»

(6) Mi -l dai.
cl_1.sg.dat cl_3.sg.acc.m give_2.sg.pres
«You give it to me.»

(7) Mi -o dai.
cl_1.sg.dat cl_3.sg.acc.f give_2.sg.pres
«You give her/it to me.»

(8) Îmi dai cartea.
cl_1.sg.dat give_2.sg.pres book_def.sg.acc
«You give me the book.»

(9) Dă -mi cartea!
give_2.sg.imp cl_1.sg.dat book_def.sg.acc
«Give me the book!»

As a general observation, syllable reduction, whether obligatory or optional, always occurs in the rightmost RWP. Obligatory sandhi to the following item, to the right, is triggered by the occurrence of an auxiliary verb starting with a vowel (e.g., *am, a, aș, oi, om*) or the 3.SG.ACC.F RWP form *o* (*mi-o* in ex. 7) .

Obligatory sandhi to the preceding item, to the left, occurs if the context for obligatory sandhi to the right is not present and the underlying form (see Section 3) of the rightmost item ends in *i* or *u*. If the preceding item is also an RWP it serves as syllabic host for the rightmost RWP (*mi-l* in ex. 6). If the rightmost RWP is the only item in the sequence it surface as *î*-prothetic form in preverbal position (*îmi* in ex. 8). In postverbal position, the verb functions as the syllabic host (*dă-mi* in ex. 9).

The RWP underlying forms *ni*, *li*, *vi* for PL.DAT exhibit a special behavior: occurring as single RWPs, thus in both the rightmost and leftmost position, they surface as their accusative counterparts *ne*, *vă*, and *le*, respectively – an instance of the aforementioned case syncretism (*le* in ex. 4). In other cluster positions, the surface forms remain identical to the underlying forms (*ni* in ex. 1).

When obligatory sandhi is not possible but specific contextual conditions are met, Romanian RWPs can undergo optional sandhi. This means that they may reduce (asyllabic *le* in ex. 11), but such a reduction is not compulsory (syllabic *le* in ex. 10). Optional sandhi to the following item, to the right, is possible if the following item starts with an unstressed vowel (ex. 11). Similarly, some RWPs in a sequence of length one can optionally attach to the preceding item, to the left, if the preceding item ends with an unstressed vowel (*să-mi* in ex. 12). In the same context, the RWP can also surface as an î-prothetic form (*îmi* in 13).

(10) Le aduci mere
cl_3.pl.dat bring_2.sg.pres apple_pl.acc
«You bring them apples.»

(11) Le- aduci mere
cl_3.pl.dat bring_2.sg.pres apple_pl.acc

«You bring them apples.»

(12) Vreau să -mi
want_1.sg.pres that cl_1.sg.dat
dai mere.
give_2.sg.pres apples
«I want you to give me apples.»

(13) Vreau să îmi
want_1.sg.pres that cl_1.sg.dat
dai mere.
give_2.sg.pres apples
«I want you to give me apples.»

3 Model description

The computational implementation of a theoretical model serves as a bridge between abstract concepts and practical applications, providing a platform to explore, validate, and extend the capabilities of the model. In linguistics, where numerous frameworks come with diverse theoretical constructs, translating these abstract models into computational terms becomes essential (see, for instance, Bender and Langendoen, 2010).

Although numerous computational tools for linguistics are available – such as Hayes et al. (2013) for OT, Kaplan et al. (2004) for LFG, Copestake (2001) for HPSG, to name a few –, none of the theoretical approaches to RWPs mentioned in Section 1 have attempted to demonstrate how these frameworks can be computationally implemented and systematically validated.

In Gerstenberger (2022), I provided comprehensive description of RWPs and their contexts of occurrence using transparently intersubjectively testable linguistics features based on their (a)syllabicity. Granted, the syllable is notoriously difficult to define in a universally agreed-upon way – is it primarily a unit of speech production (articulation), perception, or both? (cf., for instance, Ohala, 2008), but it is widely recognized as a fundamental building block in phonological theory.

Unlike Dobrovie-Sorin and Giurgea's (2013:p. 266) claim that „clitic forms are underlyingly asyllabic or syllabic", Popescu's (2003:p. 154) claim of an unspecified underlying mora, or Klein's (2007:p. 77) use of clusters with *î*-prothetic forms as model input, I propose a model in which all underlying representations are uniformly syllabic (see Table 1).

The model I propose for handling RWPs is similar to the generative approach in Chomsky and Halle (1968) and the two-level morphology in Koskenniemi (1983), as it employs two different levels of representation as well as a set of rules for mapping between the underlying and surface levels.

While the underlying representations are theoretical linguistic entities, the input forms are their computational linguistic counterparts – concrete strings that can be processed.

Given the case syncretism between accusative and dative plural RWP forms presented in Section 2 – *ni* ⇒ *ne* (syllabic) and then *ne* (syllabic) ⇒ *ne-* (asyllabic), as well as that the observation that the obligatory sandhi to the right has higher precedence than the obligatory sandhi to the left, a consecutive rule set, as presented for XFST in Beesley and Karttunen (2003), seems better suited to implement these rules.

The key idea when modeling RWP phenomena is to account for the position of a clitic in the clitic sequence: is the clitic in the rightmost position or not? Next, it is necessary to identify the correct contextual features for distinguishing between obligatory and optional sandhi. The asyllabic RWP forms are marked by vowel loss, hyphen addition, or both, while the postverbal position is characterized by the addition of hyphens between all items in the clitic sequence that follow the verb.

The following XFST grammar fragment provides definitions and rules for modeling the obligatory sandhi to the right for all RWPs that are followed by an auxiliary verb starting with a vowel or by the RWP form *o*.

```
define V_RWP "o";
define ReduceHighV_RWP "mi" | "ți" | "i" |
    "și";
define DeleteHighV_RWP "lu";
define SubstitueHighV_RWP "ni" | "vi" | "li";
define HighV_RWP ReduceHighV_RWP |
    DeleteHighV_RWP | SubstitueHighV_RWP;
define DeleteLowV_RWP "mă" | "se" | "vă";
define LowV_RWP DeleteLowV_RWP | "te" | "ne" |
     "le";
define Syllabic_RWP HighV_RWP | LowV_RWP;
define V_Initial_Aux "am" | "ai" | "a" | ...";
define C_Initial_Aux "voi" | "vei" | "va" | "
    vom" | ...;
define Aux V_Initial_Aux | C_Initial_Aux;
define Is_Obligatory_Host V_Initial_Aux |
    V_RWP;
define RWP V_RWP | Syllabic_RWP;
define Rightmost_RWP Syllabic_RWP .#. -
```

Number	Accusative					Dative					Verb *a fi*	
	1p	2p	3p.m	3p.f	3p.refl	1p	2p	3p.m	3p.f	3p.refl	1p.pres	3p.pres
Sg	/mə/	/te/	/lu/	/o/	/se/	/mi/	/tsi/	/i/	/i/	/ʃi/	/su/	/i/
Pl	/ne/	/və/	/i/	/le/	/se/	/ni/	/vi/	/li/	/li/	/ʃi/	–	/su/

Tabela 1: Underlying forms as input for the surface form generation of RWPs and RWV *a fi* «to be»

```
    Syllabic_RWP .#. Syllabic_RWP;
define Remove_Vowel
    (DeleteLowV_RWP | DeleteHighV_RWP) -> (""
    / [aeiouă] _ )
    / .#. Is_Obligatory_Host;
define Substitute_Form
    "ni" -> "ne" / _ Rightmost_RWP .#. ||
    "vi" -> "vă" / _ Rightmost_RWP .#. ||
    "li" -> "le" / _ Rightmost_RWP .#.;
define Add_Right_Hyphen Rightmost_RWP -> "-" /
     _ Is_Obligatory_Host;
define Asyllabic_Transformation Remove_Vowel
    || Substitute_Form || Add_Right_Hyphen;
regex Asyllabic_Transformation;
```

First, groups of items with the same behavior are defined. `ReduceHighV_RWP` items have the same orthographic string in both their syllabic and asyllabic forms. `DeleteHighV_RWP` and `DeleteLowV_RWP` items lose the vowel as asyllabic forms, as expressed in the rule `Remove_Vowel`. `SubstitueHighV_RWP` items change they form accordingly in the rule `Substitute_Form`, reflecting the aforementioned case syncretism.

The positional information – whether an RWP is the rightmost in the sequence – is modeled by the rule *Rightmost_RWP*: a rightmost RWP is not followed by any other RWP, which is expressed using set subtraction.

In the context of the obligatory host to the right, the rule `Add_Right_Hyphen` ensures that the correct orthographic forms are produced as surface forms.

4 Model implementation

While the XFST grammar fragment in the previous section is a proof that some linguistic phenomena described in Section 2 can be translated into XFST rules, the concrete implementation has been done in Python. There are some non-negligeable reasons for Python:

- Python allows for more expressive rule implementations.
- Python has excellent debugging and testing tools.
- Python has a rich ecosystem for NLP such as spaCy, NLTK, Stanza.
- Python syntax is cleaner and easier to read than XFST syntax.
- Python scripts can be easily documented, shared, deployed, or packaged.

The task of the surface form generation module is to produce the correct spelling of each RWP within its specific context. Since the proposed model takes syllabic RWPs as input, the following operations are performed: deleting a vowel, changing a vowel, adding a sandhi-marking hyphen, adding a prothetic *î*, and adding postverbal hyphens.

The module expects the input to provide all necessary linguistic information for each form in a given context. During the generation process, the procedure first checks the features of each item and applies the constraints on obligatory sandhi to the right, then to the left, followed by those on optional sandhi. For additional surface form adjustments, certain gerund and imperative forms may adapt to the context. If no sandhi constraints apply, the syllabic underlying RWP forms remain unchanged.

Since the main task of this project was a proof of concept – namely, to demonstrate that the linguistic description of RWPs in Gerstenberger (2022) can be implemented transparently – *spaCy*[1], an open-source software library for advanced NLP in Python was chosen for this task. Not only do *spaCy* Doc objects contain default linguistic annotations, such as part-of-speech tags and positional information of specific tokens in context, but the linguistic structure can also be adapted, extended,

[1] `https://spacy.io/`

and customized for specific types of annotation. When run on the input, all annotations can be queried and checked for contextual values, allowing operations to be performed on each token.

The input sentence (see, e.g., Figure 2) is analyzed using a «fine-tuned» *spaCy* pipeline created specifically for this test input set. As shown in the code fragment in Figure 1, the Python rules operate on each token in the spaCy Doc object.

For instance, the code checks whether the current token is a *Linear Order Part*, as defined in Gerstenberger (2007), meaning that it can occur in both preverbal and postverbal positions. If this condition is met, it further checks whether the token is an RWP or an RWV to determine whether additional operations should be performed on the input string. Along with default value checks on the current token – such as its current position and contextual elements – custom functions like `token._.is_rwp`, `token._.is_rwv`, and `is_rightmost_in_cluster(token)` have been implemented for this task. To address the dative/accusative case syncretism in the plural, an `interim_form` mapping has been employed. The entire rule set is freely available from `https://github.com/ciprian-NO/rb_rwp_generation/blob/main/generate_rwp_surface_forms.ipynb`.

To test the effectiveness of the module, I created a test input database covering all relevant RWP phenomena described in Section 2.

5 Test database creation

Evidently, the design of the test input database is driven by the RWP surface realization model proposed. The structure of a database entry is a tuple of <INPUT, {TARGET$_1$,..., TARGET$_n$}>, where the input string contains the syllabic input, i.e., the underlying representation of RWPs and the set contains all possible output variants. For example, for the curse-like expression *Înțepeni-i-s-ar toate oasele!* («May all his/her bones stiffen!»), the input is `[Înțepeni i se ar toate oasele!]`, created by removing all hyphens that indicate postverbality or asyllabicity. Furthermore, the obligatory asyllabic surface form *s* has been replaced by the syllabic underlying form *se*. Due to grapheme-phoneme ambiguities in RWPs 2, the RWP form *i* remains unchanged because it can stay both for a syllabic and an asyllabic form.

As mentioned above (see Section 2), optional sandhi refers to phonological adjustments at morpheme or word boundaries that may be influenced by factors such as individual speech patterns (idiolect), social or regional dialects (sociolect, regiolect), or the formality level of speech (speech register). As for different output targets, these are basically the result of optional sandhi, that can occur in different contexts.

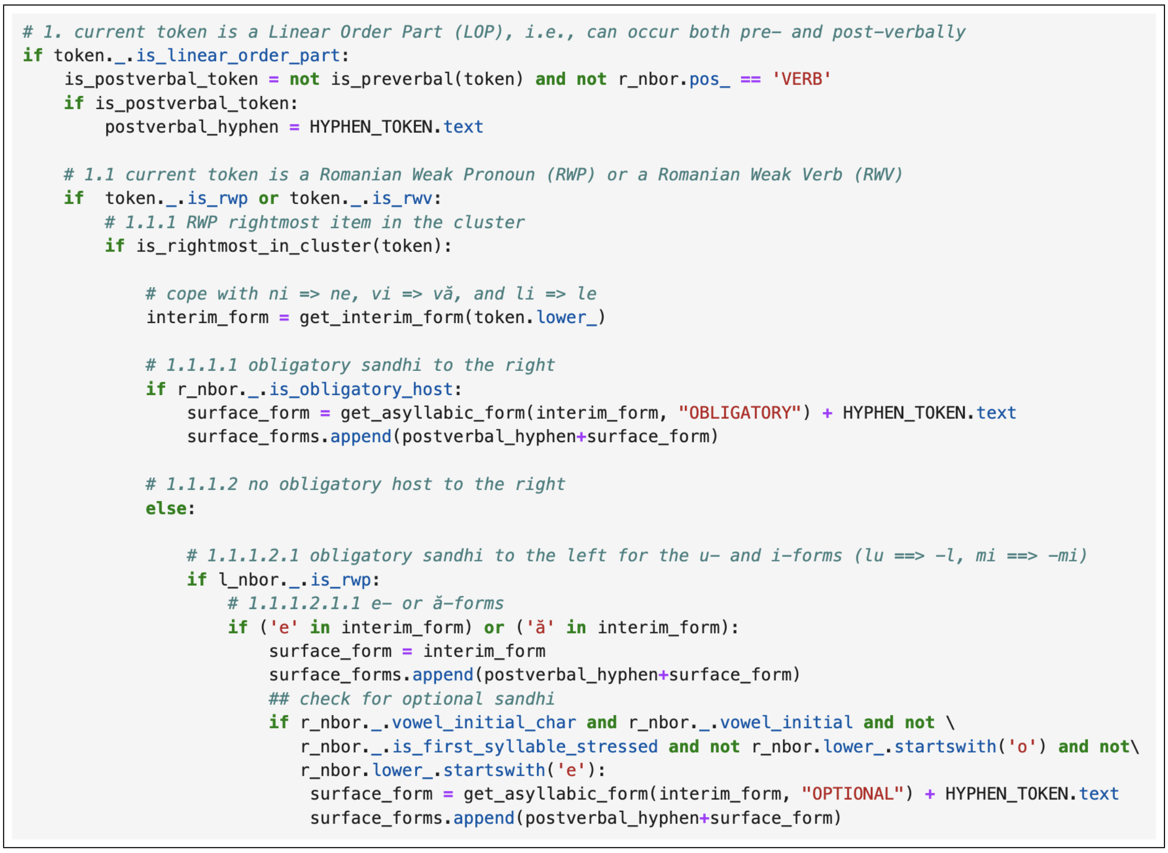
Figure 2 shows a database entry with only one target. In such an entry, the `source` string is identical to the `target` string, *Arată-ne muzeul!* («Show us the museum!»): both contain the correct surface form of the RWPs.

In database entries that contain multiple targets reflecting optional sandi, e.g., for the sentences *De ce îi dai mere?* («Why are you giving him/her apples?») in Figure 3 or *O împușcă în inimă.* («He/She shoots her in the heart.») in Figure 4, the *source* string is only one of the different targets.

RWP configurations are complex due to their syntactic positioning, morphological behavior, and interactions with other elements in the sentence. When building a specialized test input database for testing the surface realization of RWPs, several key linguistic and structural factors (see Section 2) must be considered, such as: person, number, gender, and case of RWPs, cluster length, etc.

The test input database has been primarily created based on examples from Gerstenberger (2022) and the relevant literature cited therein. RWP examples that were not complete sentences have been minimally expanded into full sentences (the *source* string in a database entry), typically by adding an adverb (e.g., *cumpără-l* «buy it» becomes *Cumpără-l repede!* «Buy it quickly!»). Some RWP sequences are repeated in different contexts to test surface realization in cases of optional sandhi.

The database contains 352 input strings and 470 target sentences, distributed as follows: 257 sets with one target output, 80 sets with two target variants, 7 sets with three variants,



```python
# 1. current token is a Linear Order Part (LOP), i.e., can occur both pre- and post-verbally
if token._.is_linear_order_part:
    is_postverbal_token = not is_preverbal(token) and not r_nbor.pos_ == 'VERB'
    if is_postverbal_token:
        postverbal_hyphen = HYPHEN_TOKEN.text

    # 1.1 current token is a Romanian Weak Pronoun (RWP) or a Romanian Weak Verb (RWV)
    if  token._.is_rwp or token._.is_rwv:
        # 1.1.1 RWP rightmost item in the cluster
        if is_rightmost_in_cluster(token):

            # cope with ni => ne, vi => vă, and li => le
            interim_form = get_interim_form(token.lower_)

            # 1.1.1.1 obligatory sandhi to the right
            if r_nbor._.is_obligatory_host:
                surface_form = get_asyllabic_form(interim_form, "OBLIGATORY") + HYPHEN_TOKEN.text
                surface_forms.append(postverbal_hyphen+surface_form)

            # 1.1.1.2 no obligatory host to the right
            else:

                # 1.1.1.2.1 obligatory sandhi to the left for the u- and i-forms (lu ==> -l, mi ==> -mi)
                if l_nbor._.is_rwp:
                    # 1.1.1.2.1.1 e- or ă-forms
                    if ('e' in interim_form) or ('ă' in interim_form):
                        surface_form = interim_form
                        surface_forms.append(postverbal_hyphen+surface_form)
                        ## check for optional sandhi
                        if r_nbor._.vowel_initial_char and r_nbor._.vowel_initial and not \
                           r_nbor._.is_first_syllable_stressed and not r_nbor.lower_.startswith('o') and not\
                           r_nbor.lower_.startswith('e'):
                            surface_form = get_asyllabic_form(interim_form, "OPTIONAL") + HYPHEN_TOKEN.text
                            surface_forms.append(postverbal_hyphen+surface_form)
```

Figura 1: Example of RWP surface generation rules implemented in Python

and 8 sets with four. The main features of the RWPs are summarized in Table 2. Even if a database with only 352 input string might seem small, it contains all relevant phenomena for handling with RWPs.

Both the input string and target sets were manually created. The input string is generated by replacing all instances of RWP surface forms with their underlying forms and removing any hyphens or prothetic *î* in the example sentence, representing an abstraction process. In contrast, creating the target set involves generating all possible variants of the input string, representing an instantiation process.

For contexts involving RWPs, all possible combinations have been created. However, in cases of optional sandhi without RWPs, only those variants covered by the rules implemented for RWP contexts have been generated. Moreover, since there is no consensus on what precisely triggers the spell-out of different variants in optional sandhi, the decision was made to include all potential variants, even though some may be more acceptable than others depending on the context.

When selecting examples for the test database, I avoided using RWP sequences that are not widely accepted, such as coordinated RWPs (see Dobrovie-Sorin and Giurgea, 2013, p. 262) or examples of RWP clitic sequences involving marginally acceptable Person Case Constraint (PCC), a constraint that restricts the co-occurrence of certain clitic pronouns, specifically based on two features: person and case (cf. Bonet, 1994).

While not aiming to exhaust all possible RWP combinations, I sought to cover all relevant phenomena described in Gerstenberger (2022). Moreover, since this database was created manually and has been incrementally corrected, it is not immune to errors, either in linguistic judgment or spelling.

As a final remark, the same input test database is intended for use in testing statistically-based generative methods, more specific, Generative Pre-Trained Transformers (GPTs). Given their widespread popularity and apparent versatility, I decided to test the GPT large language models provided by OpenAI (2024). The goal of using a shared test in-

Feature	**Preverbal forms**	**Postverbal forms**	**Example**
RWP sequence length	L1: 280 L2: 79 L3: 16	L1: 71 L2: 31 L3: 7	Postverbal L2 *Dă-mi-le acum!* «Give them to me now!»
	Obligatory î-prothetic forms	**Optional î-prothetic forms**	
Î-prothetic contexts (Only preverbal L1)	25	î-prothetic forms: 43 no î-prothetic forms: 50	Optional î-prothetic form *De ce îmi dai mere?* «Why are you giving me apples?»
	Obligatory forms	**Optional sandhi contexts**	
No *î*-prothetic contexts	267	RWP as syllabic host: 32 Context item as syllabic host: 50	Obligatory RWP forms *De ce mi le dai?* «Why are you giving them to me?»
	Syllabic context items	**Asyllabic context items**	
Optional sandhi contexts with RWP as possible syllabic host	15	17	RWP as syllabic host *N-o văd bine.* «I don't see her/it well.»
	Syllabic RWP forms	**Asyllabic RWP forms**	
Optional sandhi contexts with context item as possible syllabic host	25	25	Context item as syllabic host *El ne-arată muzeul.* «He shows us the museum.»

Tabela 2: Distribution of RWP features of target sentences in the test database

put database is to establish a foundation for fair comparison and evaluation between rule-based and GPT-based approaches.

6 Feature annotation and validation

The model relies on fine-grained linguistic information about both the currently processed item and its context, which necessitates that the input data be linguistically annotated accordingly.

As already mentioned, for processing the entire set of 352 input sentences, I chose to use *spaCy* for several reasons:

- it offers multi-language support for over 75 languages, including Romanian[2];
- it offers a hybrid approach, i.e., it supports both machine learning models and linguistic rules;
- it is highly customizable and extensible (e.g., via Custom Token Attributes such as `is_obligatory_host`.)

To address both obligatory and optional sandhi in RWP surface realization, the *spaCy* annotations have been extended with a set of functions that enrich token annotation with features for grapheme-phoneme disambiguation (`vowel_final_char` vs. `vowel_final`) and indicate whether an initial vowel is stressed (`is_first_syllable_stressed`).

Another set of functions has been devised to add high-level annotations, such as `is_rwp`, `is_rightmost_in_cluster`, or `is_obligatory_host`. These functions rely on default morphosyntactic annotations and are designed to simplify the validation of contextual features for generating the appropriate RWP form.

Finally, to ensure that hyphens indicating asyllabicity, postverbality, or both are assigned according to official writing conventions, the annotations `is_linear_order_part` and `is_preverbal` have been added to the default *spaCy* token features. In contrast to preverbal position (ex. 2), all items in a cluster – including RWPs and auxiliary verbs – are connected to the main verb by hyphens in postverbal position (ex. 3).

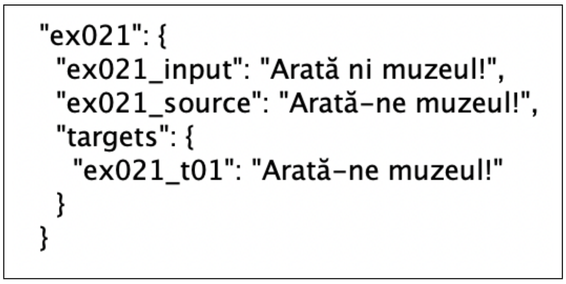
Each input string is annotated using the *spaCy* module, extended with functions for additional features, and the results are serialized in JSON format. Task-relevant annotations are then manually corrected, and finally, the JSON structures are converted to *spaCy*'s binary format. Since only task-relevant annotations are manually corrected, other annotations may contain errors.

7 Output testing

As with the development of any other handcrafted grammar, the implementation of the RWP surface generation module was an iterative process involving careful analysis, rule refinement, and extensive testing to ensure that linguistic patterns were accurately captured. Each iteration required revisiting existing rules, adding new ones, and addressing edge cases.

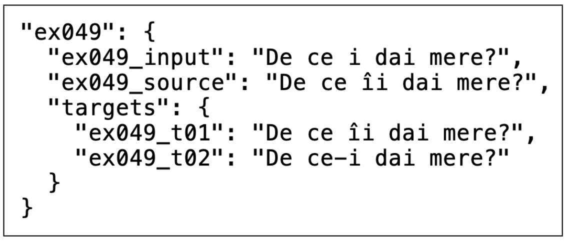
The spaCy-annotated, manually corrected

[2] `https://spacy.io/models/ro`



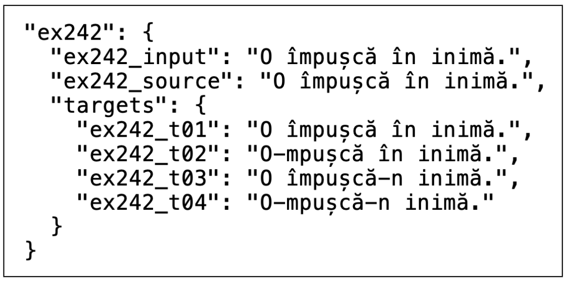
```
"ex021": {
  "ex021_input": "Arată ni muzeul!",
  "ex021_source": "Arată-ne muzeul!",
  "targets": {
    "ex021_t01": "Arată-ne muzeul!"
  }
}
```

Figura 2: Database entry with one target



```
"ex049": {
  "ex049_input": "De ce i dai mere?",
  "ex049_source": "De ce îi dai mere?",
  "targets": {
    "ex049_t01": "De ce îi dai mere?",
    "ex049_t02": "De ce-i dai mere?"
  }
}
```

Figura 3: Database entry with two targets



```
"ex242": {
  "ex242_input": "O împușcă în inimă.",
  "ex242_source": "O împușcă în inimă.",
  "targets": {
    "ex242_t01": "O împușcă în inimă.",
    "ex242_t02": "O-mpușcă în inimă.",
    "ex242_t03": "O împușcă-n inimă.",
    "ex242_t04": "O-mpușcă-n inimă."
  }
}
```

Figura 4: Database entry with four targets

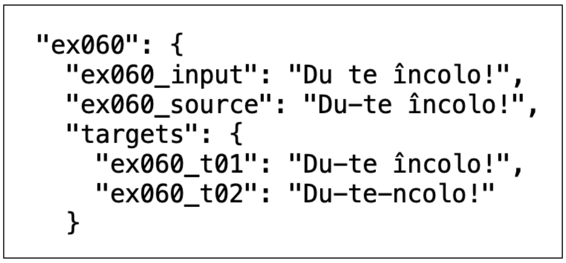
mini-corpus based on the input data is loaded and used to re-analyze the input so that the linguistic features needed for RWP surface generation are available for the application of the rules. Finally, the output is checked against the corresponding set of targets (see Figures 2 – 4).

Since this implementation is primarily a proof of concept – for immediate context checking –, I kept the processing as simple as possible. This simplicity means that the module has a stateless design, i.e., no «memory»: it does not retain information about the previously generated form of the current token, nor of the previously generated neighboring token. In standard Romanian orthography, hyphens can denote sandhi, postverbal clitics, or both. Given the lack of memory in the module, a hyphen may occasionally be generated twice.

For each token, a set of surface forms is generated based on its context. The set of output strings then comprises all possible combinations of these token surface forms. Cases of overgeneration are filtered out in the final step of processing, as illustrated by strings 3. and 4. below.

```
<060> . . . . . . . . . . . . . . . .
[4 tokens] Du te încolo !
[['Du'], ['-te', '-te-'],
 ['încolo', '-ncolo'], ['!']]
  1. ('Du', '-te', 'încolo', '!')
  2. ('Du', '-te', '-ncolo', '!')
  3. ('Du', '-te-', 'încolo', '!')
  4. ('Du', '-te-', '-ncolo', '!')
. . . . . . . . . . . . . . . . . . .
```

The remaining correct output strings 1. *Du-te încolo!* and 2. *Du-te-ncolo!* («Go away!») are checked against the targets of the corresponding database entry in Figure 5.



```
"ex060": {
  "ex060_input": "Du te încolo!",
  "ex060_source": "Du-te încolo!",
  "targets": {
    "ex060_t01": "Du-te încolo!",
    "ex060_t02": "Du-te-ncolo!"
  }
```

Figura 5: Entry with optional sandhi targets

The hand-crafted grammar is tailored specifically for a defined input set with a particular output set, making it neither bias-free in its linguistic choice of examples or evaluations nor entirely error-free in terms of spelling or implementation. Nevertheless, it represents an attempt to faithfully implement the RWP description outlined in Gerstenberger (2022) within a computational-linguistic framework.

8 Conclusions

In this article, I presented the implementation of a model for RWP surface realization, following the linguistic description provided in Gerstenberger (2022). While Gerstenberger (2018) addresses only the realization of obligatory sandhi and does not offer a computational implementation of the algorithm, this study extends the approach by incorporating both obligatory and optional sandhi within a computational framework.

A further key contribution of this work is the compilation of a systematic set of test inputs for evaluating the model, including the linguistic annotations required to meet the implemented constraints. This computational implementation serves not only as a proof of concept but also as a robust method for validating the model, demonstrating its practical applicability, and ensuring that its predictions align with empirical data. Due to its transparency and proximity to linguistic surface forms, the model is adaptable to any constraint-based linguistic framework.

Since the resources used in this study – specifically, the input test database and the surface generation module – were created manually, they may contain errors. However, these resources are freely available at `https://github.com/ciprian-NO/rb_rwp_generation` for the research community to test, improve, and expand.

References

Roxana-Maria Barbu and Ida Toivonen. 2018. Romanian Object Clitics: Grammaticalization, agreement and lexical splits. In *Proceedings of the LFG18 Conference*, page 67–87. Stanford: CSLI Publications.

Kenneth R. Beesley and Lauri Karttunen. 2003. *Finite-State Morphology.* CSLI Publications, Stanford, CA.

Emily M. Bender and D. Terence Langendoen. 2010. Computational linguistics in support of linguistic theory. *Linguistic Issues in Language Technology*, 3.

Eulàlia Bonet. 1994. The person-case constraint: A morphological approach. *MIT Working Papers in Linguistics*, 0(22):33–52.

Anca Chereches. 2014. A Prosodic Analysis of Romanian Pronominal Clitics. In *University of Pennsylvania Working Papers in Linguistics*, volume 20. Available at https://repository.upenn.edu/pwpl/vol20/iss1/7/.

Noam Chomsky and Morris Halle. 1968. *The Sound Pattern of English.* Harper & Row.

Ann A. Copestake. 2001. Implementing typed feature structure grammars. In *CSLI lecture notes series.*

Carmen Dobrovie-Sorin. 1999. Clitics across categories: The case of Romanian. In Henk C. van Riemsdijk, editor, *Clitics in the Languages of Europe.* Mouton de Gruyter.

Carmen Dobrovie-Sorin and Ion Giurgea, editors. 2013. *A Reference Grammar of Romanian*, volume 1: The noun phrase. John Benjamins.

B. Gerlach and J. Grijzenhout, editors. 2001. *Clitics in Phonology, Morphology and Syntax.* John Benjamins.

Ciprian-Virgil Gerstenberger. 2007. A mereology-based general linearization model for surface realization. In *Proceedings of EUROLAN 2007*, University of Iaşi, Romania. Available at https://www.researchgate.net/publication/233951937_A_mereology-based_general_linearization_model_for_surface_realization.

Ciprian-Virgil Gerstenberger. 2018. A grammar for romanian weak pronoun generation. In *Languages at the Crossroads: Training, Accreditation and Context of Use. Proceedings of the 35th Edition of the International Conference of the Spanish Association of Applied Linguistics*, pages 215–226. University of Jaén. Available at https://www.researchgate.net/publication/332671041_A_Grammar_for_Romanian_weak_pronoun_generation.

Ciprian-Virgil Gerstenberger. 2022. How weak are romanian clitic pronouns? *Nordlyd*, 47(1):37–57.

Bruce Hayes, Bruce Tesar, and Kie Zuraw. 2013. OTSoft 2.5 [software package]. http://www.linguistics.ucla.edu/people/hayes/otsoft/.

Ronald M. Kaplan, John T. Maxwell III, Tracy Holloway King, and Richard Crouch. 2004. Integrating finite-state technology with deep LFG grammars. In *Proceedings of the ESSLLI'04 Workshop on Combining Shallow and Deep Processing for NLP.*

Udo-Michael Klein. 2007. *Encoding of argument structure in Romanian and SiSwati.* Ph.D. thesis, University of London. Available at http://hdl.handle.net/11858/00-001M-0000-0012-8EEC-D.

Kimmo Koskenniemi. 1983. Two-level model for morphological analysis. In *IJCAI*, volume 83, pages 683–685.

Géraldine Legendre. 2001. Positioning Romanian verbal clitics at PF. In (Gerlach and Grijzenhout, 2001).

Paola Monachesi. 2001. Clitic placement in the Romanian verbal complex. In (Gerlach and Grijzenhout, 2001).

Paola Monachesi. 2005. *The Verbal Complex in Romance: A Case Study in Grammatical Interfaces.* Oxford University Press.

John J Ohala. 2008. The emergent syllable. In *The syllable in speech production.* Taylor & Francis.

OpenAI. 2024. ChatGPT [Large Language Model]. https://chat.openai.com/chat.

Alexandra Popescu. 2000. The morphophonology of the Romanian clitic sequence. In *Lingua*, volume 110, pages 773–799. Elsevier.

Alexandra Popescu. 2003. *Morphophonologische Phänomene des Rumänischen.* Ph.D. thesis, University of Düsseldorf. Available at https://docserv.uni-duesseldorf.de/servlets/-DocumentServlet?id=3187.

Kan Sasaki and Daniela Căluianu. 2000. An Optimality Theoretic Account for the Distribution of Pronominal Clitics in Romanian. Technical report, University of Tsukuba. Available at https://www.academia.edu/33668351/An_optimality_theoretic_account_for_the_distribution_of_pronominal_clitics_in_Romanian.

Stanca Somesfalean. 2007. *On the Form and Interpretation of Clitics.* Ph.D. thesis, Université du Québec à Montréal. Available at https://archipel.uqam.ca/9628/.

Oana Săvescu Ciucivara. 2009. *A Syntactic Analysis of Pronominal Clitic Clusters in Romance - The view from Romanian.* Ph.D. thesis, New York University. Available at https://www.proquest.com/docview/304954033.