

Full Inflection Learning Using Deep Neural Networks

Octavia-Maria Șulea^{1,2}, Liviu P. Dinu^{1,2}, Bogdan Dumitru^{1,2}

¹ University of Bucharest, 14 Academiei, 010014, Bucharest, Romania

²Human Language Technologies Research Center, University of Bucharest
mary.octavia@gmail.com, liviu.p.dinu@gmail.com, bdumitru@bitdefender.com

Abstract. In this paper we present a deep learning architecture capable of predicting the full inflectional paradigms from the uninflected, dictionary form of a word and the full orthographic syllabification. We use Romanian as a case study, since its morphology is rife with stem alternations (non-concatenative inflectional processes) and phonological ambiguity between hiatus and diphthongs. We show that Sequence to Sequence model receiving n-grams as input can solve this problem as good as, if not better than, the state of the art.

1 Introduction and Related Work

Knowing what form to use when learning the representation of a word in an NLP pipeline can be a challenge for languages with rich inflectional morphology[6], where certain inflected forms may be so rare within available corpora that they manifest as hapax legomenons while their corresponding uninflected form might be quite common. This frequency disparity between various forms of the same word leads to vectors of essentially the same underlying concept not ending up near each other in the embedding space, and this in turn leading to wrong conclusions of dissimilarity between the forms. The ability to automatically link all the inflected forms of a word together becomes thus crucial. To this end, research in Computational Morphology has shifted its focus from the traditional construction of rule-based morphological analyzers, which require tedious work to account for alternations in morphologically rich languages, to inflection learning and inflection extraction using machine learning and data-driven techniques.

The automatic induction of full inflections from uninflected, dictionary, forms using machine learning was studied in [11], [9], [8] and results were presented for Romanian, while the adjacent task of extracting inflection tables in a multilingual setting was introduced in [13] and closely followed by [1] and [14]. In this paper, we look at inflection learning using a deep neural network architecture. While [13] and [9] trained CRF-based models to learn transformation rules between word forms, [11], [1], and [8] used SVMs and [14] used a sequence to sequence character model based on [19]. Although we also used a sequence to

sequence model, the input into ours differed from [14] as it only uses the trailing characters representing the difference between inflected and uninflected forms, and the architecture itself differed in several points as presented below.

For inflection learning, we chose to focus on the Romanian datasets ([2] [8]) rather than on the multilingual introduced in [13] because the latter contains considerably less base forms for each language (the largest language dataset is for Finish with over 40000 base forms) than the former (over 58000 base forms). We considered that a more varied and *rich* inflection, especially in non-concatenative processes (i.e. apophony), would represent a better test for the effectiveness of a sequence to sequence architecture in inflection learning, since the dataset would give the model less opportunity to overfit particularities of the training data.

Other tasks in Computational Morphology related to inflection learning but leading to benefits in Text to Speech synthesis, *syllabification learning* or *hyphenation prediction*, were investigated as a substitute for complex rule-based systems which were shown to perform poorly in languages containing hiatus-diphthong ambiguities, such as Romanian [12]. From a high-level perspective, automatic hyphenation in ambiguous contexts tests the ability of learning models to distinguish the same sequence in different contexts, while inflection learning for morphologically rich languages tests the ability of these models to identify different sequences in similar contexts. Therefore, inflection learning and automatic syllabification can be construed as complementary tasks. In the present study, we also investigate the use of our proposed system for the task of automatic hyphenation.

2 Romanian Morphology. Alternation and Ambiguity

While Romanian has only one fully irregular verb, the verb *a fi* (to be), it is rife with partial irregularities, both in the stem and in the inflectional suffix patterns. These patterned irregularities, alternations or apophonies (ablaut), occur both in the verbal and nominal domain. Tables 1 and 2 give only a glimpse into the richness of the Romanian inflectional morphology. Extensive tables for the numerous patterns in Romanian inflectional morphology can be found in linguistic studies from [16], who identified 38 types for the verbs alone, [2], who identified 41 verb types, and [7]. As you can see in Table 1, the noun *floare* (flower) has the oa-o stem alternation, whereas the verb *a purta* (to wear) has the u-o-oa alternation (Table 2).

In the case of syllabification, Romanian, along with Portuguese [5], exhibits another interesting feature at the morpho-phonological level: the diphthong-hiatus ambiguity [4]. For instance, the character sequence "ia" (specifically [iV]) can appear in words either as a diphthong, and therefore require no hyphen during character-based syllabification (e.g. *piele* [pje.le], skin), or as two vowels in hiatus, requiring a hyphen to separate them (e.g. *biela* [bi.ela], rod). Since Romanian, unlike Portuguese, does seem to rely on consonant clusters around the vowels in hiatus and the diphthongs to make the distinction [12], we expect this context to help in automatically learning the correct syllabification.

Table 1: Alternations in the nominal domain

Tag	Regular copac (tree)	Part. Irregular floare (flower)
sg.N-A.indef.	copac	<i>floare</i>
sg.G-D.indef.	copaci	<i>flori</i>
sg.N-A.def.	copacul	<i>floarea</i>
sg.G-D.def.	copacului	<i>florii</i>
pl.N-A.indef.	copaci	<i>flori</i>
pl.G-D.indef.	copaci	<i>flori</i>
pl.N-A.def.	copacii	<i>florile</i>
pl.G-D.def.	copacilor	<i>florilor</i>

Table 2: Alternations in the verbal domain

Tag	Regular a visa (to dream)	Irregular a fi (to be)	Part. Irregular a purta (to wear)
1st sg.	vis-ez	sunt	<i>port-</i>
2nd sg.	vis-ezi	ești	<i>porț-i</i>
3rd sg.	vis-ează	este	<i>poart-ă</i>
1st pl.	vis-ăm	suntem	<i>purt-ăm</i>
2nd pl.	vis-ați	sunteți	<i>purt-ați</i>
3rd pl.	vis-ează	sunt	<i>poart-ă</i>

3 Approach

3.1 Dataset and Features

The datasets we used for verbal and nominal inflection learning as well as for hyphenation were based on [3]. In the verbal domain, we extracted over 7000 entries with full inflections and in the nominal domain, around 51000 entries. We split the dataset into test and validation with an 80-20 ratio, for each task. Several prepping steps were applied to the input data, performing proper character alignment, n-gram vocabulary extraction, and padding. One hot encoding was used as a final step in representing the input.

We tried several different feature inputs including character n-grams. The one that performed best for inflection learning was the pair of trailing characters representing the difference between inflected and uninflected forms. For instance, for the verb *a purta* (to wear), the second person singular model was trained on the pair (*urta*, *orți*) from *purta* (wear) and *porți* (you.SG wear).

For nouns, we initially observed 91 form labels, when taking into consideration gender tags also. This was due to our dataset containing versions of the same nominal base form for two or all three of the genders: masculine, feminine, and neuter. This in turn lead to a lot of variation in the gender tags. We decided to eliminate the gender tags for nouns from the labeling and this way we obtained 8 forms:

1. singular, nominative-accusative, indefinite
2. singular, nominative-accusative, definite
3. singular, genitive-dative, indefinite
4. singular, genitive-dative, definite
5. plural, nominative-accusative, indefinite
6. plural, nominative-accusative, definite
7. plural, genitive-dative, indefinite
8. plural, genitive-dative, definite

For the verbal domain, we focused on the present indicative tense, which is known to be the most irregular [11]. Here, a more intuitive 6 label set emerged:

1. first person, singular
2. second person, singular
3. third person, singular
4. first person, plural
5. second person, plural
6. third person, plural

For the hyphenation learning task, the preprocessing implied extracting character level n-grams ($n=6$) over the orthographically syllabified words. This generated $n+\text{len}(\text{word})$ vectors which were labeled as either 1, if the middle of the n-gram contained a hyphen, and zero otherwise.

3.2 Deep Learning Approach

For inflection learning, we used the *sequence to sequence* (*seq2seq*) model presented in [19] since this type of model has enjoyed great success in a variety of ML tasks requiring a mapping from one sequence to another. Sequence to sequence learning implies training a model to convert sequences from one domain to sequences from another domain. In our case, we convert the uninflected form of a verb and noun to an inflected form. This is achieved by coupling a so-called encoder with a decoder, feeding the dictionary form to the encoder and expecting the decoder to produce the correct inflected form. For the encoder, a LSTM layer was used to map input data to a fixed sized vector. The resulted vector was then passed to the decoder, implemented using again an LSTM layer to extract the sequence for the vector. A resizing layer was used between encoder and decoder. To improve performance, a proper alignment was used for both input and output data.

For hyphenation, we used an architecture similar to [15]. We implemented these models using Keras¹, wrapping over the Python library for deep learning, TensorFlow². We also used SciPy, NumPy, and sk-learn [18] for the preprocessing steps.

For each inflection learning task and each form, we trained a separate network capable of learning the uninflected-inflected pairing. Therefore, we had 8

¹ <https://keras.io>

² <https://www.tensorflow.org/>

trained binary models for the nominal domain, and 6 for the verbal, learning good and bad uninflected-inflected character contexts. The results presented below represent an average of the performance of the 8 and 6 models respectively.

4 DNN Architecture Description

There are several ways to implement a seq2seq model. In our case we have selected a gated RNN layer, more specifically an LSTM, to implement both the encoder and decoder of the model.

At a high level we can view an *RNN* as a function:

$$y_n = RNN(x_{1:n})$$

where $x_i \in \mathbb{R}^l$ and $y \in \mathbb{R}^m$

The *RNN* function is defined recursively by a function *R* that is using a state vector s_{i-1} and x_i as input vector to produce the next state vector s_i . Then, the vector s_i is mapped to y_i using another function *O*.

We can rewrite the *RNN* in terms of *R* and *O* as follows:

$$\begin{aligned} RNN^*(x_{1:n}, s_0) &= y_{1:n} \\ y_i &= O(s_i) \\ s_i &= R(s_{i-1}, x_i) \end{aligned}$$

where $x_i \in \mathbb{R}^l$, $y \in \mathbb{R}^m$ and $s_i \in \mathbb{R}^m$

An LSTM is a more complex RNN since it has a gated architecture. In fact, it is the first to implement a gating mechanism. When the LSTM was designed, its main functionality was to prevent the vanishing gradient problem from happening.

Using the above formalism we can now define an LSTM:

$$\begin{aligned} s_j &= R(s_{j-1}, x_j) = [c_j, h_j] \\ c_j &= f \odot c_{j-1} + i \odot z \\ h_j &= o \odot \tanh(c_j) \\ i &= \sigma(x_j W^{xi} + h_{j-1} W^{hi}) \\ f &= \sigma(x_j W^{xf} + h_{j-1} W^{hf}) \\ o &= \sigma(x_j W^{xo} + h_{j-1} W^{ho}) \\ z &= \tanh(x_j W^{xz} + h_{j-1} W^{hz}) \\ y_j &= O(s_j) = h_j \end{aligned}$$

where $s_j \in \mathbb{R}^{2l}$, $x_i \in \mathbb{R}^m$, $c_j, h_j, i, f, o \in \mathbb{R}^l$, $W^{xo} \in \mathbb{R}^{lm}$ and $W^{ho} \in \mathbb{R}^l$ and i, f, o are gates controlling the input, forgetting, and output. The state vector s_j is split in two vectors c_j and h_j , the first representing the memory component and the second, the hidden state. Gates values are a linear combination between the input x_j and the previous state h_{j-1} . Figure 1 explains this.

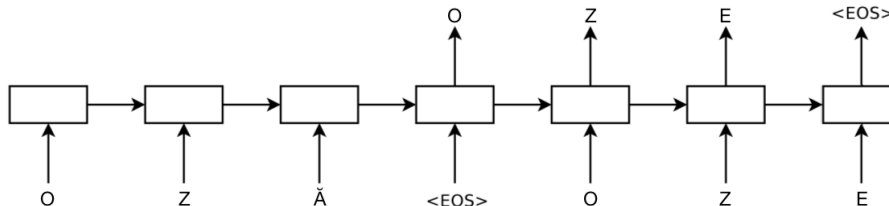


Fig. 1: Out model reads 'OZÄ' sequence (from *poza* - "picture") and produces 'OZE' (from *poze* "pictures") as output sequence. The Seq2seq model stops when the space character is generated.

5 Results

Table 1 shows the results of our model in the verbal and nominal domain and in the syllabification task. For the nominal inflection learning task as well as for hyphenation, our system surpasses the state of the art, while for the verbal inflection learning task our system's performance is on par with the system presented in [9]. For inflection learning, we also compare our results with those in [14] obtained on the Finnish dataset, which, although smaller than our own, it is the closest in number of verb base forms and inflection paradigms. The noun inflection learning results from [14] are given as a rough comparison, since their dataset contained a magnitude less nominal base forms than ours.

As we can see, the results show that our system brings significant improvement in the nominal domain and in automatic hyphenation and is on par, with a slight edge, with the state-of-the-art for conjugation learning.

One thing none of the inflection learning systems do is to fully generate the inflected form starting from the uninflected form. For example, our system learns that for the verb *a purta* (to wear) one of its forms, *poartă*, has a u-oa stem alternation and a ta-tă ending transformation, however it was not designed to generate the full inflected form. This is an issue also for the systems published in previous works ([10], [11], [9], [8], [14]), with only [14] and [8] attempting to move in semi-supervised directions, but not achieving better results than the supervised approaches.

Table 3: Accuracy results of our model compared to previous work

Model	Task	Acc.
Seq2Seq	Verbal	98.7%
[11]	Verbal	90.64%
[9]	Verbal	98.5%
[14]	FI-V	97.97%
Seq2Seq	Nominal	99%
[8]	Nominal	83.15%
[14]	FI-NA	95.44%
Seq2Seq	Hyphenation	99.5%
[12]	Hyphenation	95%

6 Conclusions

In this paper we showed that a sequence to sequence model can be used as a morphological analyzer of uninflected forms and is able to learn full inflections of both verbs and nouns in a morphologically rich language, Romanian. We’ve also investigated its use in learning end of the line hyphenation, or orthographic syllabification for Romanian. For hyphenation and nominal inflection learning our system outperforms the state of the art, while for verbal inflection learning, it matches the performance in [9].

Acknowledgments

We would like to thank the reviewers. The work of Liviu P. Dinu was supported by UEFISCDI, project number 53BG/2016.

References

1. Ahlberg, M., Forsberg, M., Hulden, M.: Paradigm classification in supervised learning of morphology. In: Mihalcea et al. [17], pp. 1024–1029, <http://aclweb.org/anthology/N/N15/N15-1107.pdf>
2. Barbu, A.M.: Conjugarea verbelor românești. Dicționar: 7500 de verbe românești grupate pe clase de conjugare. Bucharest: Coresi (2007), 4th edition, revised. (In Romanian.) (263 pp.)
3. Barbu, A.M.: Romanian lexical databases: Inflected and syllabic forms dictionaries (2008)
4. Chitoran, I.: The phonology of Romanian. A constraint-based approach. Mouton de Gruyter (2002)
5. Chitoran, I., Hualde, J.I.: From hiatus to diphthong: the evolution of vowel sequences in romance. *Phonology* 24, 37–75 (2007)
6. Cotterell, R., Schütze, H.: Morphological word-embeddings. In: Mihalcea et al. [17], pp. 1287–1292, <http://aclweb.org/anthology/N/N15/N15-1140.pdf>

7. Şulea, O.M.: Alternations in the Romanian verb paradigm. Analyzing the indicative present. Master's thesis, Faculty of Foreign Languages and Literatures, University of Bucharest (2012), available at <http://ling.auf.net/lingbuzz/001562>
8. Şulea, O.M.: Semi-supervised approach to romanian noun declension. In: Knowledge-Based and Intelligent Information Engineering Systems: Proceedings of the 20th International Conference KES-2016, York, UK, 5-7 September 2016. pp. 664–671 (2016)
9. Dinu, L.P., Şulea, O.M., Niculae, V.: Sequence tagging for verb conjugation in romanian. In: RANLP. pp. 215–220 (2013)
10. Dinu, L.P., Ionescu, E., Niculae, V., Şulea, O.M.: Can alternations be learned? A machine learning approach to verb alternations. In: Recent Advances in Natural Language Processing 2011. pp. 539–544 (September 2011)
11. Dinu, L.P., Niculae, V., Şulea, O.M.: Learning how to conjugate the romanian verb. Rules for regular and partially irregular verbs. In: European Chapter of the Association for Computational Linguistics 2012. pp. 524–528 (April 2012)
12. Dinu, L.P., Niculae, V., Şulea, O.M.: Romanian syllabication using machine learning. In: TSD. pp. 450–456 (2013)
13. Durrett, G., DeNero, J.: Supervised learning of complete morphological paradigms. In: Vanderwende, L., III, H.D., Kirchhoff, K. (eds.) Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA. pp. 1185–1195. The Association for Computational Linguistics (2013), <http://aclweb.org/anthology/N/N13/N13-1138.pdf>
14. Faruqui, M., Tsvetkov, Y., Neubig, G., Dyer, C.: Morphological inflection generation using character sequence to sequence learning. CoRR abs/1512.06110 (2015), <http://arxiv.org/abs/1512.06110>
15. Fritzke, B., Nasahl, C.: A neural network that learns to do hyphenation. In: International Joint Conference on Neural Networks (1991)
16. Guţu-Romalo, V.: Morfologie Structurală a limbii române. Editura Academiei Republicii Socialiste România (1968), in Romanian
17. Mihalcea, R., Chai, J.Y., Sarkar, A. (eds.): NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015. The Association for Computational Linguistics (2015)
18. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830 (Oct 2011)
19. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. CoRR abs/1409.3215 (2014), <http://arxiv.org/abs/1409.3215>