ISSN 1870-4069

Spanish Morphology Generation with Deep Learning

Carlos Escolano, Marta R. Costa-jussà

Universitat Politècnica de Catalunya,
TALP Research Center,
Barcelona

{carlos.escolano, marta.ruiz}@tsc.upc.edu

Abstract. Morphology generation is the natural language processing task of generating word inflection information. In this paper, we propose a new classification architecture based on deep learning to generate gender and number in Spanish from non-inflected words. This deep architecture uses a concatenation of embedding, convolutional and recurrent neural networks. We obtain improvements compared to other standard machine learning techniques. Accuracy of our proposed classifiers reaches over 98% for gender and over 93% for number.

Keywords: Morphology, deep learning, spanish.

1 Introduction

Morphology generation consists in generating the inflection information of a word. For example, given a lemmatized sentence like *El casa pequeño* in Spanish, a morphological generator would output *La casa pequeña*. Generally speaking, a morphological simplified text allows to reduce vocabulary specially in highly inflected languages. There are many natural language processing tasks and other related applications that can benefit from morphology generation to boost its performance. There have been many works in morphological generation and some of them are in the context of the application of machine translation.

Most of the works for this application translate to a morphologically simplified target, and then use some morphological generation technique to find the final output. To name a few, for example, [15] build maximum entropy markov models for inflection prediction of stems; [3, 10] use conditional random fields (CFR) to predict one or more morphological features; and [6] use Support Vector Machines (SVMs) to predict verb inflections.

A different but related task is the one of Part-of-Speech (PoS) tagging which aims at labelling words with its corresponding syntactic role. For this task, the form of the word is not simplified, so the form itself contains much more information than in the task of morphology generation. In this field the number of works is huge, but most related works to our own would be: [8] where authors train a model to predict each individual fragment of a PoS tag by means of machine learning algorithms; [4] where authors propose a deep learning architecture for English PoS tagging; and [11] where authors perform fine-grained PoS tagging with feedforward

Table 1. Examples of text representations.

Model	Text
Original	La casa de la playa
PoS & Lemma	DA0FS0[el] NCFS000[casa] SPS00[de] DA0FS00[el] NCFS000[playa]
Simplified Text	DA00[el] NC000[casa] SPS00[de] DA00[el] NC000[playa]
Simplified-Gender	DA0S0[el] NCS000[casa] SPS00[de] DA0S0[el] NCS000[playa]
Simplified-Number	DA0F0[el] NCF000[casa] SPS00[de] DA0FS[el] NCF000[playa]

and bidirectional recurrent architectures. In this paper, we focus on the task of generating morphological attributes for a particular task and we propose a deep learning architecture which concatenates embedding, convolutional and recurrent neural networks. This architecture is particularly tested on generating number and gender for the Spanish language.

The inputs to our system are lemmas and their corresponding fine-grained PoS tag where information of number and gender has been removed. Given the nature of our architecture, it could be further generalized to generating the fine-grained PoS tag from lemmas and for any language. The rest of the paper is organised as follows. Section 2 describes the morphology generation architecture. Section 3 details the experimental framework and section 4 reports accuracy of the classifier compared to other state-of-the-art techniques. Finally, section 5 highlights main achievements of this work.

2 Morphology Generation Architecture

This section reports which is the input data to our task, it describes which is the architecture proposed and it explains the motivation of this architecture.

2.1 Input Data

To train our system we start from a Spanish corpus and we use a morphological analyzer to remove the information of gender and number. Table 1 shows an example. Given our simplified text in gender and number, we propose to train two different models: one to retrieve gender and another to retrieve number. Each model decides among three different classes. Classes for gender classifier are masculine (M), femenine (F) and none (N); and classes for number classifier are singular (S), plural (P) and none (N).

2.2 Description

Inspired by previous Collobert's work [4], as features for our classifier we use windows of words as input. Each word is represented by a fixed size window of words in which the central element is the one to classify.

Figure 1 shows an example of windows of length 3. Note that in the example "de" does not have a window assigned because this word is invariant in gender and number. Following the example, our classifiers do not have to train all types of words. Some types of words, such as prepositions (*a, ante, cabo, bajo, de...*), do not have gender or

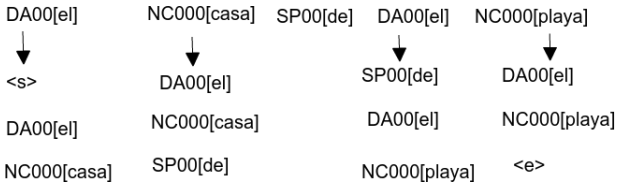

Fig. 1. Text to window representation.

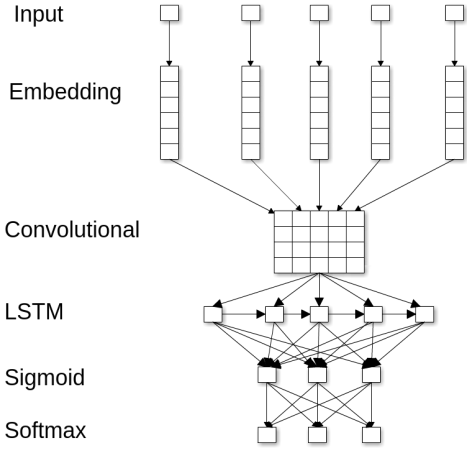

Fig. 2. Neural network overview.

number. Therefore our system was trained for determiners, adjectives, verbs, pronouns and nouns which are the ones that present morphology variations in gender or number. However, note that all types of words are used in the windows.

We base our architecture also in Collobert's proposal [4] and we modify it by adding a recurrent neural network. This recurrent neural network is relevant because it keeps information about previous elements in a sequence and, in our classification problem, context words are very relevant.

As a recurrent neural network, we use a Long Short Term Memory (LSTM) [9] that is proven to be efficient to deal with sequence NLP challenges [14]. This kind of recurrent neural network is able to maintain information for several elements in the sequence and to forget it when needed. Figure 2 shows an overview of the different layers involved in the final classification architecture, which are detailed as follows:

Embedding. We represent each word as its index in the vocabulary, i.e. every word is represented as one discrete value:

$$E(w) = W, W \in R^d. \tag{1}$$

w being the index of the word in the sorted vocabulary and d the size of the array. Then, each word is represented as a numeric array and each window is a matrix. This process is trained as part of the neural network architecture.

Table 2. Corpus details.

	Sentences	**Words**
Training	58,688	2,297,656
Development	990	43,489
Test	1010	44,306

Convolutional. We add a convolutional neural network. This step allows the system to detect some common patterns between the different words. This layer's input consists in a matrix W^l of multidimensional arrays of size $n \cdot d$, where n is the window length (in words) and d is the size of the array created by the previous embedding layer. This layer's output is a matrix of the same size as the input.

Max Pooling. This layer allows to extract most relevant features from the input data and reduces feature vectors to half.

LSTM. Each feature array is treated individually, generating a fixed size representation h_i of the ith word using information of all previous words (in the sequence). This layer's output, h, is the result of the last element of the sequence using information from all previous words.

Sigmoid. This layer smoothes results obtained by previous layer and compresses results to the interval $[-1, 1]$. This layer's input is a fixed size vector of shape $1 \cdot n$ where n is the number of neurons in the previous LSTM layer. This layer's output is a vector of length c equal to the number of classes to predict.

Softmax. This layer allows to show results as probabilities by ensuring that the returned value of each class belongs to the $[0, 1)$ interval and all classes add up 1.

2.3 Motivation

The input data of the classification algorithm is morphogically simplified in terms of gender and number. This simplification largely reduces the information that can be extracted from individual words in the vocabulary. In addition, we can encounter out-of-vocabulary words for which no morphological information can be extracted.

The main source of information are the context words. The information of a word consists in itself and the words that surround it (a window of words). Sometimes relevant information preceeds the word and sometimes information is after the word. Words (like adjetives), which are modifying or complementing another word, generally take information from preceeding words.

For example, in the sequence *casa blanca*, the word *blanca* could also be *blanco, blancos or blancas* but because noun and adjective are required to have gender and number agreement, the femenine word *casa* forces the femenine for *blanca*. While, for example, determiners usually take information from posterior words. This fact motivates that the word to classify has to be placed at the center of the window.

Finally, the fact that we rely only on the context information (since words themselves may not have any information) makes the recurrent neural network a key element in our architecture. The output h of the layer can be considered a context vector of the whole window maintaining information of the previous encountered words (in the same window).

Table 3. Distribution of the gender classes in the corpus.

Set	Word Type	Total	Femenine(%)	Masculine(%)	None(%)
Train	Determiners	340.739	53,24	38,36	8,40
	Nouns	571.053	52,19	46,52	1,29
	Verbs	219.638	14,24	12,75	73,01
	Pronouns	43.806	4,28	6,91	88,81
	Adjetives	185.107	21,69	24,14	54,17
Development	Determiners	6.534	51,18	41,20	7,62
	Nouns	11.025	51,39	46,78	1,83
	Verbs	5.630	11,10	13,20	75,70
	Pronouns	1.079	3,89	6,12	89,99
	Adjetives	3.129	60,95	38,69	0,36
Test	Determiners	6.858	52,07	40,76	7,17
	Nouns	11.347	51,72	46,56	1,72
	Verbs	4.629	12,18	13,23	74,59
	Pronouns	1.015	4,24	7,39	88,37
	Adjetives	3.375	24,68	24,21	51,11

3 Experimental Framework

This section reports the dataset used for experimentation together with its preprocessing. We also detail which are the parameters used for the final architecture configuration.

3.1 Data, Preprocessing and Software

As data, we use a subset of the United Nations Corpus [13]. Corpus statistics are shown in Table 2. Corpus preprocessing consisted in tokenization and lowercasing. PoS tagging was done using *Freeling* [12]. All chunking or name entity recognition was disabled to preserve the original number of words.

With the *Freeling* information, we represent each word with its PoS tag and lemma. From each PoS tag, we remove gender and number information to create the simplified text representation. For example, the word *La* becomes *DA0FS0[el]*, where *F* indicates its gender (femenine) and *S* its number (singular). This word is simplified to *DA00[el]*. A consequence of this new representation is that the determiners *el, la, los, las* are all represented as *DA00[el]* which introduces additional ambiguity to the task. See Table 1 for a full example of text simplification.

We do not consider for classification all word types. On the one hand, words that do not have explicit morphology according to *Freeling*'s tagset are not classified. However, these words are still used in the windows of words as context relevant information. On the other hand, word types being classified are *determiners, nouns, pronouns, adjectives, and verbs*, as shown in Tables 3 and 4. Balance details in gender and number for the different sets is detailed in Tables 3 and 4, respectively.

To generate the classification architecture we used the library *keras* [2] for creating and ensambling the different layers. Using NVIDIA GTX Titan X GPUs with 12GB of memory and 3072 CUDA Cores, each classifier has a training time of aproximately

Table 4. Distribution of the number classes in the corpus.

Set	Word Type	Total	Singular(%)	Plural(%)	None (%)
Train	Determiners	340.739	61,80	38,19	0,01
	Nouns	571.053	67,75	31,92	0,33
	Verbs	219.638	41,04	28,46	30,50
	Pronouns	43.806	12,21	8,09	79,70
	Adjetives	185.107	63,38	36,38	0,24
Development	Determiners	6.534	61,75	38,25	0
	Nouns	11.025	66,93	32,68	0,39
	Verbs	5.630	42,38	27,06	30,56
	Pronouns	1.079	9,55	7,04	83,41
	Adjetives	3.129	60,95	38,69	0,36
Test	Determiners	6.858	60,95	39,05	0
	Nouns	11.347	65,51	34,14	0,35
	Verbs	4.629	40,53	29,66	29,81
	Pronouns	1.015	9,66	9,16	81,18
	Adjetives	3.375	58,34	41,45	0,21

Table 5. Values of the different parameters of the classifiers.

Parameter	Gender	Number
Window size	7	9
Vocabulary size	7000	9000
Embedding	128	128
Filter size	5	7
LSTM nodes	70	70

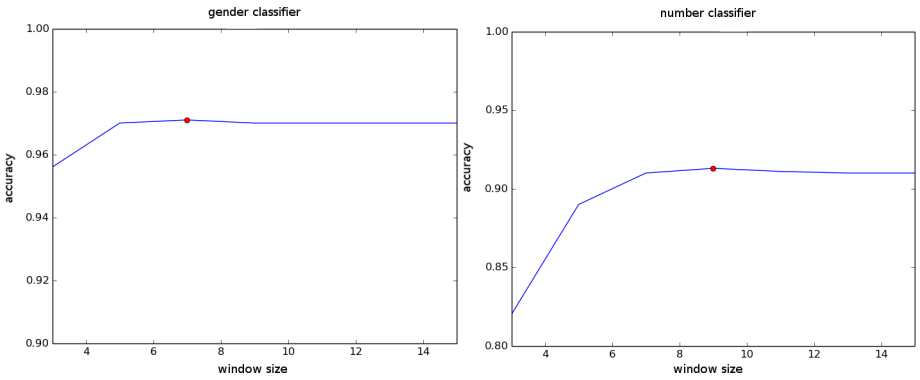

Fig. 3. Window size.

1h calculating two epochs of the model with batch size 32 and optimizer *adadelta* with default parameters.

3.2 Parameters Details

Regarding classification parameters, experimentation has shown that number and gender classification tasks have different requirements.

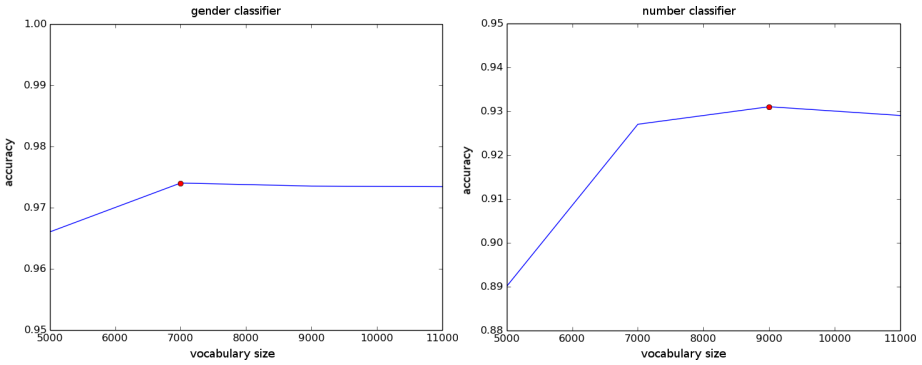

Fig. 4. Vocabulary size.

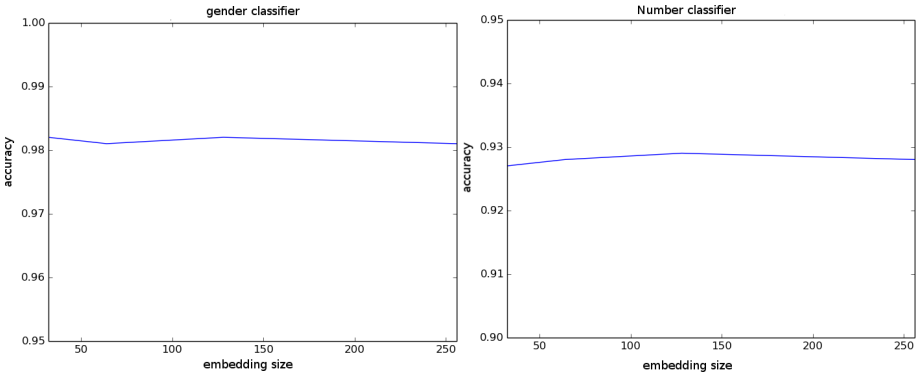

Fig. 5. Embedding size.

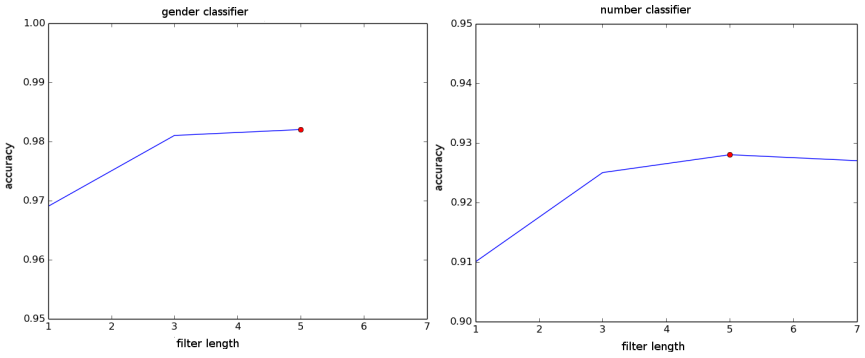

Fig. 6. Filter size.

Table 5 summarizes these parameters and details are given as follows.

– The best size of the window is found in 7 and 9 words for gender and number respectively. In both cases (number and gender) increasing this size lowers the accuracy of the system as shown in Figure 3.

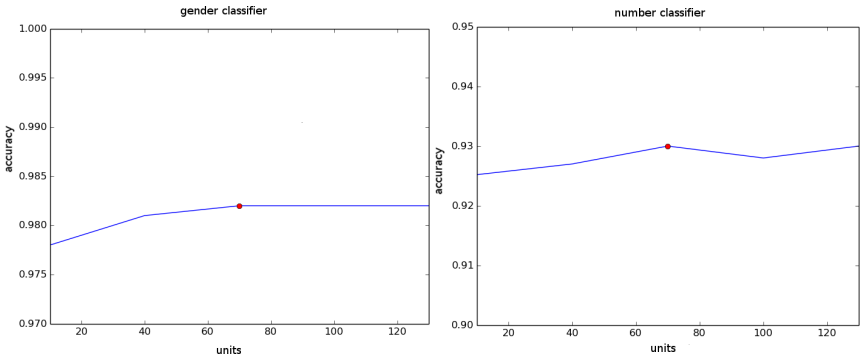

Fig. 7. LSTM nodes.

– The vocabulary size is a trade-off between giving enough information to the system to perform the classification while removing enough words to train the classifier for unknown words. We fix values to 7,000 and 9,000 for gender and number, respectively. The accuracy curve varying the vocabulary size is shown Figure 4).

– An embedding size of 128 results in stable training, while further increasing this value augmented the training time and hardware cost without impact in accuracy (see Figure 5). The filter size in the convolutional layer produced the best results when it was slightly smaller than the window size, being 5 and 7 the best values for gender and number classification, respectively (see Figure 6).

– Increasing LSTM nodes up to 70 improved significantly for both classifiers (see Figure 7).

4 Evaluation

Table 6 shows results for the classification task both number and gender. We have contrasted our proposed classification architecture based on neural networks with standard machine learning techniques such as linear, cuadratic and sigmoid kernels SVMs [5], random forests [1], convolutional[7] and LSTM[9] neural networks (NN). All algorithms were tested using features and parameters described in previous section with the exception of random forests in which we added the one hot encoding representation of the words to the features. We observe that our proposed architecture achieves by large the best results in all tasks.

5 Conclusions

This paper shows a new deep learning architecture for morphology generation. Our task is challenging because number and gender are generated from a word without this inflection. Our architecture uses several layers including embedding, convolutional and recurrent, which are able to outperform state-of-the-art techniques such as support vector machines or other well-known deep learning architectures.

Table 6. Classification results. In bold, best results.

Algorithm	**Accuracy**	
	Number	**Gender**
Naive Bayes	61.3	53.5
Lineal kernel SVM	68.1	71.7
Cuadratic kernel SVM	77.8	81.3
Sigmoid kernel SVM	83,1	87.4
Random Forest	81.6	91.8
Convolutional NN	81.3	93.9
LSTM NN	68.1	73.3
CNN + LSTM	**93.7**	**98.4**

We are able to improve accuracy almost by absolute 5% for gender classification and over 10% for number compared to convolutional neural networks and SVMs, respectively, which are the second-best performing systems. We reach over 98% accuracy for gender and over 93% accuracy for number. Further work includes using our architecture for PoS tagging and integrating it in a machine translation system.

Acknowledgments. This work is supported by the Spanish Ministerio de Economía y Competitividad and European Regional Development Fund, through the postdoctoral senior grant *Ramón y Cajal* and the contract TEC2015-69266-P (MINECO/FEDER, UE).

References

1. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
2. Chollet, F.: Keras: Deep learning for humans (2015), https://github.com/fchollet/keras
3. Clifton, A., Sarkar, A.: Combining morpheme-based machine translation with post-processing morpheme prediction. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp. 32–42 (2011)
4. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. Journal of Machine Learning Research 12, 2493–2537 (2011)
5. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (1995)
6. Formiga, L., Costa-jussà, M.R., Mariño, J.B., Fonollosa, J., Barrón-Cedeño, A., Màrquez, L.: The TALP-UPC phrase-based translation systems for WMT13: System combination with morphology generation, domain adaptation and corpus filtering. In: Proceedings of the Eighth Workshop on Statistical Machine Translation. pp. 134–140 (2013)
7. Fukushima, K.: Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics 36(4), 193–202 (1980)
8. Giménez, J., Màrquez, L.: SVMTool: A general POS tagger generator based on support vector machines. In: Proceedings of the 4th International Conference on Language Resources and Evaluation. pp. 43–46 (2004)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation 9(8), 1735–1780 (1997)

10. Kholy, A.E., Habash, N.: Rich morphology generation using statistical machine translation. In: Proceedings of the Seventh International Natural Language Generation Conference. pp. 90–94 (2012)
11. Labeau, M., Löser, K., Allauzen, A.: Non-lexical neural architecture for fine-grained POS tagging. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 232–237 (2015)
12. Padró, L., Stanilovsky, E.: Freeling 3.0: Towards wider multilinguality. In: Proceedings of the Language Resources and Evaluation Conference (LREC'12). pp. 2473–2479 (2012)
13. Rafalovitch, A., Dale, R.: United Nations general assembly resolutions: A six-language parallel corpus. In: Proceedings of Machine Translation Summit XII: Posters. pp. 292–299 (2009)
14. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Proceedings of the 27th International Conference on Neural Information Processing Systems. vol. 2, pp. 3104–3112 (2014)
15. Toutanova, K., Suzuki, H., Ruopp, A.: Applying morphology generation models to machine translation. In: Proceedings of the conference of the Association for Computational Linguistics and Human Language Technology (ACL-HLT). pp. 514–522 (2008)