Homonymy Resolution and Tagging of Grammar Classes without Lexical Information using Neural Networks*

J. Ernesto Gómez-Balderas[1], Grigori Sidorov[1], Héctor Jiménez-Salazar[2]

[1]Center for Research in Computer Science,
National Polytechnic Institute,
Av. Juan Dios Batiz, s/n, Zacatenco, 07738,
Mexico City, Mexico
[2]Autonomous University of Mexico,
Mexico City, Mexico
sidorov@cic.ipn.mx, josegoba@sagitario.cic.ipn.mx,
hgimenez@gmail.com

Abstract. The paper describes the experiments that use artificial neural networks for resolution of the homonymy of parts of speech using limited contextual information, namely, only the information of parts of speech of the context is provided. We describe several experiments using backpropagation neural networks. Our experiments are conducted for Spanish language using LEXESP corpus. Two types of experiments are presented: the first one, when we are not aware of the real possible homonymy, and, thus, have to choose one of the fifteen possible grammar classes (in case of Spanish); and the second one, when we rely on the information obtained from the morphological analyzer, and resolve only the real homonymy presented in the text. For the first case we obtained more than 46% of precision, and for the second case more than 77%. Unlike other tagging methods we do not rely on lexical information. The advantage of this approach is drastic reduction of dimensionality of the decision space.

1 Introduction

The problem of part of speech tagging is one of the well developed problems in the area of computational linguistics. The problem consists in assigning of grammar tags to words in the text. There are different types of grammar tags, for example, they can include only parts of speech (grammar classes) information like nouns, verbs, etc. They can also include more detailed grammatical information. The part of speech tagging is very important in a lot of applications such as access to textual data base, robust syntactic analysis, etc.

Our purpose is to find a supervised method for tagging and resolution of morphological homonymy (it is equivalent to tagging with the subset of tags) that

* The work was done under partial support of National Polytechnic Institute, Mexico (CGPI, COFAA, PIFI) and Mexican government (CONACyT, SNI).

Advances in Artificial Intelligence and Applications
Research in Computer Science 32, 2007, pp. 305–314

Received 15/06/07
Accepted 31/08/07
Final version 22/10/07

depends on distributional properties of tags in the text and does not depend on lexical information.

In this work, we apply the neural networks paradigm to POS tagging. Two types of the experiments are presented: the first one, when we are not aware of the real possible homonymy, and, thus, have to choose one of the fifteen possible grammar classes for Spanish; and the second one, when we rely on the information obtained from the morphological analyzer [8], and resolve only the real homonymy presented in the text, usually there are two or three tags for homonymic words.

The paper has the following structure. First, we briefly discuss the existing tagging methods, then we describe the preparation of the input data and experimental procedures for both cases (full tagging or homonymy resolution). After this, we present experimental results for various network structures and various weight assignments to different networks. Finally, conclusions are drawn.

2 POS Tagging Methods

There are different machine learning techniques that are used in POS tagging. For example, POS tagging with Hidden Markov Model (HMM) ([15]) involves counting occurences of the sequences and building a table of their probabilities. In advanced HMM learning, these probabilities are counted for triplets and larger sequences. Some other important current trends in POS tagging include Viterbi algorithm, Brill tagging algorithm, and Baum-Welch algorithm ([5], [6], [7], [10], [12], [13], [14]). All these approaches are dependent on lexical information, i.e., on coocurrences of individual words.

In [11], the distributional technique is applied to the similar formulated problems. The reported results for English have similar precision to ours, about 50%. In Spanish, the tagging problem using context is more difficult due to free word order.

On the other hand, neural networks ([1], [2], [4]) are also used in natural language processing. Martin Eineborg and Björn Gambäck ([3]) describe experiments with three different architectures of backpropagation networks for recognition of unknown words. Veronis and Ide ([16]) were using neural network approach for extracting lexical information from dictionaries. Another suggestion ([9]) is to use word structure for tagging with neural networks. Nakamura *et al.* ([17]) proposed prediction of words categories using neural network architecture called NETgram. This was also backpropagation approach. The grammatical category of preceding word was used to predict the next word category. We substantially extend these experiments.

3 Methodology of the Experiments

3.1 Input Data Preparation

The input data are obtained from the LEXESP corpus developed in Barcelona Polytechnic University. We had part of this corpus with manually resolved

homonymy, while the whole corpus is marked with all possible morphological tags, i.e, the homonymy is not resolved.

The first step was finding in the original corpus the phrases that were disambiguated manually (they were not subsequent phrases). Thus, we obtain the gold standard of the Spanish POS tagging. This data is used while calculating precision in automatic way.

The information in the corpus is encoded using the standard *Eagles* that is *de facto* encoding standard for Spanish language. For example, the word *alegre* (*happy*) has the following encoding: *AQ0CS0*, where *A* stands for adjective (also can be *N* for noun, *V* for verb, etc.), *Q* for qualifying, *C* for common (i.e., it can be both masculine and feminine), and *S* stands for singular.

At the next step, we prepared the corpus that contains only tags of grammar classes, i.e., the first letter of the code, as presented in Table 2 and Table 3. In our case, we always use only the first element of the tag that corresponds to the grammar class, say, *A* for adjective, and write it to a derived corpus of pure tags.

Finally, we prepared the input for neural networks that should contain only 1 and 0, because this is the only possible input of the networks. There are fifteen grammar classed in Spanish that we considered. Each tag from this file has an associated binary code containing only one 1 in the position of the tag in the ordering list, e. g., 001000000000000.

3.2 Neural Networks Training

We created eight networks depending on the number of context words. The difference between these networks is the number of the input neurons, see Table 1. Each network has fifteen output neurons that contain the final weights calculated for each grammar category. The number of input neurons depends on the number of contextual words (more specifically, their grammar categories) that are taken into account. Namely, it is the number of context words multiplied by number of possible grammar categories (fifteen in our case), for example, if we have two context words, then there are 30 input neurons, etc.

We used 1, 2 or 3 hidden layers in the neural networks in different experiments, see Table 4.

The following steps are made for training of these networks applying backpropagation approach.

Step 1:

Initialize weights connections with random values.

Step 2:

Present input patterns from the corpus. Depending on the context, the word was sent to the input of one or several networks. The networks were configured according to Table 1, where *P* is the desired tag in training and the unknown tag in recognition, and tags C_n correspond to tags of the context.

Step 3:

Calculate actual output for all networks.

Calculate input for neurons of the hidden layers on the basis of values calculated for the input neurons. For hidden neuron j, we have:

$$net_{p,j} = \sum_{i=1}^{N} w_{ji} x_{pi}$$

Calculate output for hidden neurons.

$$y_{p,j} = f_j(net_{p,j})$$

Calculate output for output neurons.

$$net_{p,k} = \sum_{j=1}^{L} w_{k,j} y_{p,j}$$

$$y_{p,k} = fk(net_{p,k})$$

Step 4:

Calculate the error for all neurons, starting from the output layer.

If k is a neuron from the output layer then the value is calculated as:

$$\delta^{o}_{p,k} = (d_{p,k} - y_{p,k}) f'^{o}_{k}(net^{o}_{p,k})$$

If neuron j is not from output layer then:

$$\delta_{p,j} = x_{p,i}(1 - x_{p,i}) \sum_{k} \delta^{o}_{p,k} w^{o}_{k,j}$$

Step 5:

Update weights starting from the output layer and going back to the input layer adjusting weights.

For weights of neurons in the output layer:

$$w^{o}_{k,j}(t+1) = w^{o}_{k,j}(t) + \Delta w^{o}_{k,j}(t+1)$$

$$\Delta w^{o}_{k,j}(t+1) = \alpha \delta^{o}_{p,k} y_{p,j}$$

For weights of neurons in the hidden layer

$$w_{j,i}(t+1) = w_{j,i}(t) + \Delta w_{j,i}(t+1)$$

$$\Delta w_{j,i}(t+1) = \alpha \delta_{p,j} x_{p,i}$$

Step 6:

This process is repeated until the mean error of the cycle is less than 0.001. $Ep = \frac{1}{2} \sum_i (d_i - o_i)^2$, where d_i is the element i of the desired output and o_i is the element i of the actual output.

3.3 Experimental Procedure for Tagging

The data from the LEXESP subcorpus, for which we have the gold standard, was divided into phrases. They were introduced into all possible networks, depending on the context of each phrase, e. g., if the phrase have thee words, and we are working with the word in the middle, then the networks 0, 1 and 3 participate in the processing, see Table 1.

For training, we introduced the tags of the context words into the corresponding neural networks and compared the results with the desired tag propagating the error backwards.

Table 1. Networks structure.

Phrase structure	Network
P-C_1	0
C_1-P	1
P-C_1-C_2	2
C_1-P-C_2	3
C_1-C_2-P	4
C_1-P-C_2-C_3	5
C_1-C_2-P-C_3	6
C_1-C_2-P-C_3-C_4	7

where P is a word we are working with, and C_n stands for the context words.

Table 2. Example of the input data.

Subphrases	Desired tag	Network
_V	Y	0
_VN	Y	2
_N	V	0
Y_	V	1
_NS	V	2
Y_N	V	3
Y_NS	V	5
_S	N	0
V_	N	1
_SN	N	2
V_S	N	3
YV_	N	4
V_SN	N	5
YV_S	N	6
YV_SN	N	7

where "_" stands for the tag that is currently processed (it is considered as unknown). It can be seen that the subphrase *YVN* was input for three networks 0, 1, 3 with input data *_V, Y_, Y_N*.

As usual in application of neural networks, the recognition is the same procedure only without propagation of error backwards. After training, we pass to experiments. In each phrase, we deleted one tag for experimenting, but its value was kept for comparison; this procedure was repeated for each tag. The network guesses correctly if the kept tag value was equal to the tag obtained in the network output. If more than one network was used, then we count the average value. We also tried different weights for different networks, see Table 5, Table 6, and Table 7.

A detailed example of the procedure is presented in Table 2. Suppose that have the tagged phrase *YVNSN*. It will be introduced into networks in the manner presented there.

The same phrases were used as input data for Brill tagger and TnT tagger for comparison with the proposed approach, see below.

3.4 Experimental Procedure for Morphologic Homonymy Resolution

We have at our disposal all possible tags for cases of morphological homonymy that occur in our corpus, as well as the gold standard of tagging, i.e., manual resolution of this homonymy. Our task is to choose one tag of the reduced set of possible tags. The difference with the previous case is that we do not have to use all tags, but only the tags that are homonymic in the real text. Namely, instead of fifteen tags, we have to choose between two or three possible tags.

The procedure is essentially the same as described in the previous section, being the only difference the restricted set of tags. For example, the homonymic tags are presented in Table 3.

Table 3.Homonymic tags.

Correct tags	Other possible tags
V	
T	N
N	TV
.	.
T	VM
N	

4 Experimental Results

In this section, we present experimental results obtained using previously described procedures.

4.1 Results for Tagging

We used 797,698 entries in the experiments. We experimented with the values of parameters like number of layers, number of iterations and learning factor for neural networks. The results presented in Table 4 were obtained, being the best performance 46.61%.

We compared our results with the results obtained using Brill tagger and TnT tagger. We understand that both taggers are not designed for the task as we formulate it, but, on the other hand, let us imagine that the language has only fifteen words (that corresponds to grammar categories), so the taggers that work with words can be applied as well. For Brill tagger, we obtained precision of 15.5%, and for TnT tagger only 6%. These are very low values that we explain with lack of information for correct performance of both taggers.

Table 4. Results using different network parameters.

Iterations number	Number of hidden layers	Learning factor	Precision %
250	3	0.25	46.38
150	3	0.25	46.30
150	2	0.5	**46.61**
100	3	0.25	42.80
100	2	0.25	46.37
100	2	0.5	46.01
100	2	0.75	46.10
100	2	0.9	45.78
100	1	0.9	46.28
100	1	0.5	46.16

4.2 Results for Morphologic Homonymy Resolution

The experiment for resolution of morphological homonymy was conducted for 1,830,349 entries that were used as input for different networks depending on the context size. We obtained precision of 77.08% using networks without any additional weighting, i.e., the results of all networks were considered equally important.

Still, another hypothesis is possible, that some contexts are more important than the others. For verifying this, we assign different weights to different networks at the moment of calculation of the resulting mean value. We assign weights in the following manners:

- Symmetric weights.
- Greater weights to the networks with major left contexts.
- Greater weights to the networks with major right contexts.

The results are presented in the following tables (Table 5, Table 6, Table 7). As can be observed, a preference to the right contexts gives a little bit better results, 77.45%.

Table 5. Results for symmetric weights.

Neural network weights								Precision
R0	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**%**
0.3	0.3	0.5	0.5	0.5	0.8	0.8	1	**77.36**
0.2	0.2	0.3	0.3	0.3	0.5	0.5	0.8	76.89
0.5	0.5	0.7	0.7	0.7	0.8	0.8	0.9	77.34
1	1	0.8	0.8	0.8	0.5	0.5	0.4	76.62
1	1	0.8	0.5	0.5	0.8	1	1	76.62

Table 6. Results for preference to the left contexts.

Neural network weights								Precision
R0	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**%**
0.2	1	0.2	0.5	1	0.5	1	1	75.71
0.5	1	0.5	0.8	1	0.8	1	0.8	**76.88**
0.5	1	0.3	0.5	1	0.2	1	0.5	76.45
0.5	1	0.3	0.5	1	0.2	0.8	0.5	76.38
0.5	1	0.5	0.5	1	0.5	0.5	0.5	76.53
0.2	1	0.2	0.5	1	0.2	0.5	0.5	75.70

Table 7. Results for preference to the right contexts.

Neural network weights								Precision
R0	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**%**
1	0.2	1	0.5	0.2	0.8	0.5	0.5	76.55
1	0.2	1	0.5	0.2	1	0.5	1	76.37
1	0.2	1	0.8	0.2	0.8	0.5	0.8	76.74
1	0.1	1	0.6	0.1	0.8	0.6	0.8	76.23
0.8	0.1	0.8	0.6	0.1	0.6	0.6	0.6	76.67
1	0.2	1	0.3	0.2	0.5	0.2	0.5	75.67
0.4	0.4	0.6	0.6	0.6	1	1	1	**77.45**

5 Conclusions

We presented experiments that use artificial neural networks for resolution of the homonymy of parts of speech (grammar classes) using the limited contextual information, namely, only the information of parts of speech of the context is provided.

We describe several experiments using backpropagation neural networks. Our experiments are conducted for Spanish language using LEXESP corpus.

Two types of experiments are presented: the first one, when we are not aware of the real possible homonymy, and, thus, have to choose one of the fifteen possible grammar classes (in case of Spanish); and the second one, when we rely on the information obtained from the morphological analyzer, and resolve only real homonymy presented in the text. For the first case we obtained more than 46% of precision, and for the second case more than 77%.

We tried various network architectures and various ways of weighting of different networks.

The advantage of the approach that does not rely on lexical information is drastic reduction of dimensionality of the decision space.

In future, it is interesting to calculate precision values for each grammar class separately.

References

[1] Kishan Mehrotra, Chilukuri K. Mohan, Sanjay Ranka. *Elements of Artificial Neural Networks*. The MIT Press, 1997.

[2] Russell D. Reed and Robert J. Marks. *Neural Smithing*. The MIT Press, 1999.

[3] Martin Eineborg and Björn Gambäck. "Tagging Experiments Using Neural Networks". In: R. Eklund (Ed.) *Proceedings of the 9th Scandinavian Conference on Computational Linguistics*, Stockholm University, Stockholm, Sweden, June 1993, pp. 71-81.

[4] B. Müller and J. Reinhardt. *Neural Networks*. Springer-Verlag, 1990.

[5] Eric Brill. Some Advances in Transformation-Based Part of Speech Tagging. In: *Proceedings of the twelfth national conference on Artificial intelligence* (vol. 1), USA, 1994, pp. 722-727.

[6] Atro Voutilainen. A syntax-based part of speech analyzer. In: *Proc. of 7th Conf. of European Chapter Assoc. Comp. Linguistics*, ACL, 1995, pp 157–164.

[7] Tetsuji Nakagawa and Taku Kudo and Yuji Matsumoto. Revision Learning and its Application to Part of Speech tagging. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics* (ACL), Philadelphia, USA, July 2002, pp. 497-504.

[8] Francisco Velásquez, Alexander Gelbukh, Grigori Sidorov. AGME: Un Sistema de Análisis y Generación de la Morfología del Español. In: Julio Gonzalo, Anselmo Peñas, Antonio Ferrández (Eds.) *Proc. Multilingual Information Access and Natural Language Processing, International Workshop (November 12) at IBERAMIA-2002, VII Iberoamerican Conference on Artificial Intelligence*, Sevilla, Spain, November 12-15, 2002, p. 1–6.

[9] Helmut Schmid. Part-Of–Speech Tagging with neural networks. In: *Proceeding of COLING-94*, 1994, pp 172-176.

[10] Qing Ma, Masaki Murata, Kiyotaka Uchimoto, Hitoshi Isahara. Hybrid Neuro and Rule-Based Part of Speech Taggers. In: *Proceedings of the 18th International Conference on*

Computational Linguistics (COLING'2000), Saarbrucken, Germany, August, 2000, pp.509-515.

[11] Hinrich Schütze. Distributional Part-of-Speech Tagging. In: *Proceedings of the seventh conference on European chapter of the Association for Computational Linguistics*, Ireland, 1995, pp 141-145.

[12] Andrei Mikheev. Learning Part-of-Speech Guessing Rules from Lexicon: Extension to Non-Concatenative Operations. In: *Proc. of the 16th COLING*, Denmark, 1996, pp 770-775.

[13] Peter A. Heeman, James F. Allen. Incorporating POS Tagging into language modeling. In: *Proc. Eurospeech '97*, Rhodes, Greece, 1997, pp. 2767-2770.

[14] Simon Cozens, Pusey Lane. Primitive Part-Of-Speech Tagging using Word Length and Sentential Structure. *CoRR: Computation and Language*, 1998.

[15] Thorsten Brants. TnT A Statistical Part-Of-Speech Tagger. In: *Proceedings of the 6th Applied NLP Conference, ANLP-2000*, USA, April 29 -- May 3, 2000.

[16] Jean Veronis and Nancy Ide. Word Sense Disambiguation with Very Large Neural Networks Extracted from Machine Readable Dictionaries. In: *Proc. of Int. Conf. COLING*, Finland, 1990, pp. 1-6.

[17] M. Nakamura *et al.* Neural network approach to word category prediction for English texts. In: *Proc. of 13th Int. Conf ACL*, Finland, 1990, pp. 213-218.