Ensemble Romanian Dependency Parsing with Neural Networks

Radu Ion, Elena Irimia, Verginica Barbu Mititelu
Research Institute for Artificial Intelligence, "Mihai Drăgănescu"
13 "Calea 13 Septembrie", Bucharest 050711, Romania
{radu, elena, vergi}@racai.ro

Abstract

SSPR (Semantics-driven Syntactic Parser for Romanian) is a neural network ensemble parser developed for Romanian (a Python 3.5 application based on the Microsoft Cognitive Toolkit 2.0 Python API) that combines the parsing decisions of a varying number (in our experiments, 3) of other parsers (MALT, RGB and MATE), using information from additional lexical, morpho-syntactic and semantic features. SSPR outperforms the best individual parser (MATE in our case) with 1.6% LAS points and it is in the same class with the top 5 Romanian performers at the CONLL 2017 dependency parsing shared task. The train and test sets were extracted from a Romanian dependency treebank we developed and validated in the Universal Dependencies format. The treebank, used in the CONLL 2017 Romanian track as well, is open licenced; the parser is available on request.

Keywords: parsing, treebank, neural networks.

1. Introduction

A reliable, fast and freely available syntactic parser for Romanian was needed (by both linguists and computer scientists working in Natural Language Processing) when we decided to develop SSPR. Even if other attempts at creating Romanian parsers were documented before, none proved to be accessible or performant enough: Călăcean and Nivre (2009) reported a dependency parser based on MaltParser (Nivre et al., 2007) and trained on a small treebank (Hristea and Popescu, 2003), whose impressive results, 88.6% labelled attachment score (LAS) and 92% unlabelled attachment score (UAS), are expectable given the shortness and syntactic simplicity of the sentences in the data set; the Fips constituency rule-based parser was adapted for Romanian (Seretan et al., 2010) and is available for online use (http://www.latl.unige.ch/), but no evaluation is provided for this language; Colhon and Cristea (2016) focus only on noun phrases, using patterns to identify relations inside such structures; the dependency parser of Perez et al. (2016) reported 78.04% LAS when trained on an 8000 sentences treebank. Dumitrescu et al. (2017) RBG-based parser obtained a LAS of 79.44% in the CONLL 2017 Romanian track but new developments of Boroș and Dumitrescu (2018) employ a Bidirectional LSTM network that is still training at the time of the writing and whose foreseen results are much better.

In a nutshell, the SSPR ensemble parser we have developed works by training a feed-forward neural network (NN) to recognize "good" vs. "bad" dependency relations from the minimum spanning tree (MST) of the directed graph of all existing (as many as possible) parses for the same sentence. For the training part, each edge of the graph is weighted by the inverse of the size of the set of parsers that found that edge (the "majority voting" combination strategy). The aim of the SSPR ensemble parser is to improve on the baseline "majority voting" combination of dependency parses by using the additional, lexical, morpho-syntactic and semantic features of the words participating in a relation, coupled with information related to the state of the partially completed parse.

2. Working data

The train/dev/test data that the SSPR NN used to train and evaluate came from the Romanian treebank annotated in Universal Dependencies (CONLL-U) format, named RoRefTrees (Barbu Mititelu et al., 2016b). It is a relatively large treebank (9523 sentences, 218365 words) with complex sentences (23 tokens/sentence on average) and a large coverage: it contains several text genres (literature - 1818 sentences, law - 1606 sentences, medical - 1210 sentences, FrameNet translations - 1092 sentences, academic writing - 950 sentences, news - 933 sentences, science - 362 sentences, Wikipedia - 251 sentences, miscellanea - 1302 sentences). The corpus, consistently annotated and manually validated, is freely available as a release at universaldependencies.org and it was used as training/development/testing data for Romanian in the CONLL 2017 dependency parsing shared task. Additionally, it can be queried online with user-friendly tools: http://clarino.uib.no/iness/page?page-id=iness-main-page, http://bionlp-www.utu.fi/dep_search, http://lindat.mff.cuni.cz/services/pmltq/#!/home.

3. Features

3.1 Linguistic Features

SSPR started as a project that aimed to elude errors typical to statistical parsing by employing additional semantic information. The standard practice in machine learning/statistical approaches to parsing is to use *morpho-syntactic* (POS and/or morpho-syntactic descriptions) and *lexical* (lemmas/word forms of the words and word embeddings computed from large raw corpora) features of the words to be linked and of those in a context (*n*-word windows) around the targeted words. Also, *first-order syntactic* features (extracted directly from the dependency relations themselves) and *second-order syntactic* features (the partial analysis of a word, when already linked in the partial tree) complete the linguistic information usually exploited for automatic parsing. However, additional features, whenever available, are easy to integrate in data-driven systems and our initial intuition was that *semantic* features, like wordnet relations, sub-categorization frames and semantic classes can increase parsing performance. These features were provided by external programs and supplementary resources (e.g. an inventory of subcategorisation frames of verbs occurring in RoRefTrees, semi-automatically developed and hand validated by linguists). Moreover, information about constituency of noun, verb, adjectival and adverbial phrases was available through TTL (Ion,

2007), a tool that does the tokenization, lemmatisation and POS-tagging of the corpus.

The set of linguistic features that SSPR used to improve the "majority voting" classifier is:

- Sets of semantically-related words, derived by automatically clustering word embeddings (the feature *SemClass*) extracted from the CoRoLa corpus (Tufiș et al., 2016);
- Manually validated subcategorisation frames for all the main verbs in the treebank, enriched with semantic restrictions on the arguments (Barbu Mititelu et al., 2016a) (the features *VFrame*, identifying the frame, and the *VArg*, identifying the associated argument(s), restricted to specified Romanian WordNet (Tufiș and Barbu Mititelu, 2014) senses);
- Lexical chains between the words in the sentence (Ion and Ștefănescu, 2009): a lexical chain between a word w_1 and a word w_2 is a path in the Romanian WordNet that seeks to get from a specific sense of the word w_1 to the relevant sense of the word w_2 by following semantic relations in wordnet (the feature *LXC*);
- Morpho-syntactic features provided by TTL (the features *Type*, *Gender*, *Case*, etc.);
- Continuous constituents provided by TTL (the feature *Chunk*).

3.2 The individual parsers and their performances

The following open-source parsers were trained and evaluated using all the available linguistic features (see the scores marked with "+") and using only the standard, lexical and morpho-syntactic features: 1) MALT parser with the LIBSVM classifier and with the "arc-eager" algorithm; 2) RBG parser with the standard, projective analysis model (Lei et al., 2014); 3) MATE parser with "default" parameters (Bohnet, 2010).

Parsers performance was evaluated in a standard, 10-fold cross validation manner (with 80% training data, 10% development data and 10% test data) and Table 1 shows the average scores on the 10 different runs.

The evaluation measures we employed were: 1) **UAS**, which gives the number of relations identified in the test set (the head and the dependent are both correctly identified), but it is not concerned with the relation type (subject, object, etc.); 2) **LAS**, which also considers the dependency relations label; 3) **UAS/LAS without punct (UASp, LASp)**, which refers to the scores UAS and LAS as defined above, but ignoring the dependency relation that links the punctuation (**punct**, in our relation inventory); 4) **Relaxed LAS (LASr)**, which refers to LAS as defined above, but ignoring the relations specific to the Romanian treebank, like "nmod:tmod", which is a temporal noun modifier. This relation is an adaptation of the "nmod" relation, universally defined within the Universal Depenedency project (de Marneffe et al., 2014), i.e. valid for all natural languages. The project allows for language-specific relations, but only as subtypes of the universal ones and labelled accordingly. Thus, "nmod:tmod" is a language-specific relation, a subtype of the "nmod" relation, namely a temporal "nmod" (nominal modifier). All the relations of the type "universal:particular" are automatically reduced to "universal"; in our example, if the parser detects the relation "nmod", it is classified as correct when compared to "nmod:tmod"; 5) **Relaxed LAS without punct (LASpr)**, which is the relaxed LAS score, as defined above, that also ignores the dependency relation that links punctuation (it is the most permissive LAS score of all).

Table 1 shows little but consistent improvement of all the parsers performances when using the additional syntactic-semantic features over the standard scenarios (see LAS+ vs. LAS, UAS+ vs. UAS, etc.). We underlined the best values for each score and MATE parser superior yield is visible for all measures, except UASp+ (where the difference to the best parser's performance is insignificant). It is important to note that we could obtain these improvements in the various UAS/LAS scores of *all the three* parsers only when using *all* of our supplementary features. Otherwise, there are certain subsets that work well for some parsers but degrade performance for other parsers. We thus think that it is important to optimize the supplementary feature set according to the parsers one wants to combine, such that the supplementary feature set provides improvements for all the parsers of the ensemble.

Score	MALT	RBG	MATE
LAS	0.7978	0.7906	0.809
LAS+	0.8038	0.8091	0.8138
UAS	0.8544	0.8603	0.8668
UAS+	0.859	0.8693	0.8707
UASp	0.8639	0.8706	0.8758
UASp+	0.8683	0.8808	0.88
LASp	0.7992	0.7913	0.8096
LASp+	0.8054	0.8116	0.8152
LASr	0.813	0.805	0.8229
LASr+	0.8185	0.8225	0.8276
LASpr	0.8165	0.8077	0.8256
LASpr+	0.822	0.8271	0.8309

Table 1: The contribution of supplementary syntactic-semantic features to the UAS and LAS average scores for MALT, RBG and MATE parsers.

4. SSPR implementation

SSPR is a feed-forward neural network (NN) that learns to recognize a "correct" syntactic dependency relation vs. an "incorrect" one, from the MST "Majority Voting" combination strategy of multiple parses, taking into account its context as modelled by linguistic and structural features. There is no limit on the number of different parsers that can be combined by this NN, but it is of utmost importance that the parsers are as different as possible, from the analysis algorithm point of view, so that the reunion of all the dependency relations of a

sentence, generated by all available parsers (i.e. the directed graph of all existing parses of the sentence), should cover the correct analysis of the sentence. This way, we can make sure the combination algorithm will have a pool of choices from which to produce a correct analysis.

SSPR is a Python 3.5 application that uses Microsoft Cognitive Toolkit 2.0 Python API to train and run the NN whose structure is presented in Figure 1. The input vector *X* is a binary vector and has N positions (corresponding to N values for all the features available for one training example, i.e. a dependency relation). There are two hidden layers, the first with 100 neurons (*H*), and the second with 10 neurons (*G*). The activation function for both hidden layers is *tanh*. The structure was obtained experimenting with NNs having 0, 1, or two hidden layers, and with 10, 50, 100, 500 or 1000 neurons in a hidden layer. The NN in Figure 1 was the one that gave the best results. The NN outputs a real vector of two positions $\hat{Y}$, which is a probability distribution of the two possible states (correct/incorrect), computed with the *softmax* function.

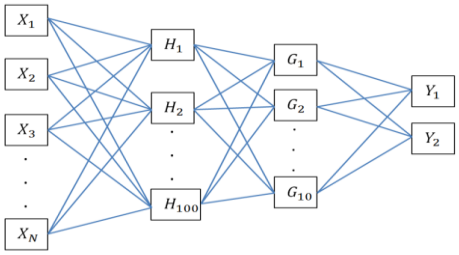

Figure 1: SSPR neural network

The training examples for SSPR NN are acquired as follows:

- **Step 1.** For each sentence in the training set, all the individual parsers were run and a directed graph A was constructed with the reunion of all the resulting analyses. The nodes of the graph A are lexical units, while the edges are dependency relations. Each edge is labelled with the name of the corresponding dependency relation and with the name of the parser that discovered it;
- **Step 2.** If a relation *r* is identified between the words *HEAD* and *DEP* by more than one parser, then all the edges between *HEAD* and *DEP* labelled *r* are replaced by a single edge, which bears the label *r* and the set V of the parsers that discovered *r*;
- **Step 3**. From the directed graph A, a Minimum Spanning Tree (MST) is constructed, using Chu-Liu-Edmonds algorithm (Chu & Liu, 1965; Edmonds, 1967) where the weight of each edge is 1/|*V*|. In other words, we computed the tree that covers all the sentence and whose dependency relations are discovered simultaneously by as many of the individual parsers as possible (i.e. the real number 1/|*V*| is minimised). Steps 1-3 are reproducing the "Majority Voting" (MV, Surdeanu and Manning, 2010) strategy of combining independently-trained parsing models;
- **Step 4.** The MST constructed at step 3 contains both correct and incorrect relations (given that the natural language may pose complicated parsing problems that none of the parsers available knows how to solve). Each relation in this tree is transformed in a training example *(X, Y)* for the SSPR NN, where *X* is the binary feature vector (see below) and *Y* is a binary vector with 2 positions computed by checking if the relation is also present in the gold-standard train set, which guarantees its correctness (*Y*= [1; 0] for a correct relation and [0; 1] for an erroneous one).

For each dependency relation that has to be classified as correct or incorrect, we construct and then concatenate in the input vector *X* the following "one-hot" vectors: 1) The binary vector of the parsers that identified the relation; 2) The binary vector of the morpho-syntactic properties of the dependent in the relation; 3) The binary vector of the label of the relation; 4) The binary vector of the dependent's lemma; 5) The binary vector of the syntactic-semantic features of the relation; 6) The binary vectors obtained by concatenating the vectors obtained at steps 1-5 for the regent of the relation, for the leftmost child of the dependent and for the regent of the regent.

SSPR NN is then trained to minimise the average (over the train set) of the cross entropy of the *Y* and $\hat{Y}$ vectors, standing for the ground-truth value and the estimated value with *softmax*.

At run-time, the SSPR parser ensemble works as follows:

- **Step 1.** All individual (the same as in training) parsers are run on a new test sentence and the directed graph A of all individual analyses is constructed;
- **Step 2.** The NN is applied on each edge a_i in A and the probability vector is computed as $\hat{Y}$ = [*P*(*correct*|*X*); *P*(*incorrect*|*X*)];
- **Step 3.** A Minimum Spanning Tree is constructed on the graph A, but this time the weight of each edge is given by the first position in the vector Y as computed by NN, namely by *P*(*correct*|*X*).

In this manner, SSPR tries to improve on the MV combination strategy by proposing more informed weights for the MST construction.

5. SSPR Evaluation

To demonstrate the improvements that the SSPR parser brings in comparison with the MV ensemble baseline, we first compute a breakdown of the LAS score of the MV ensemble as a function of the number of individual parsers that find the correct dependency relation, averaged over our 10-fold cross validation test run. Table 2 shows the components of LAS for our three parsers MV ensemble, namely the LAS when all three parsers agree on the correct dependency relation (see the "3/3" row in Table 2) and the LAS when two out of three parsers agree on the correct dependency relation ("2/3"). When only one

parser finds the correct dependency relation ("1/3"), its LAS, in this case, contributes to the "oracle" LAS score, which is the maximum possible score that can be achieved by any ensemble of these three parsers.

3/3 LAS+	**0.7218**
2/3 LAS+	**0.0968**
MV LAS+	**0.8186**
1/3 LAS+	**0.068**
ORACLE LAS+	**0.8866**

Table 2: A breakdown of the LAS score as a function of how many parsers found the correct dependency relation

Table 3 shows the SSPR scores compared to the MATE scores, the best individual parser from Table 1.

Score	MATE average	SSPR average
LAS+	**0.8138**	**0.83**
UAS+	**0.8707**	**0.8844**
UASp+	**0.88**	**0.8928**
LASp+	**0.8152**	**0.8307**
LASr+	**0.8276**	**0.8434**
LASpr+	**0.8309**	**0.8461**

Table 3: Comparison between MATE and SSPR (averages over 10 train/dev/test random samples)

The evaluation setting of SSPR is the same as the one described in Section 3.2. The individual parsers were trained on the train set, using all the available syntactic-semantic features, and were run on the development set. The neural network SSPR was trained on the development set combining the three individual parsers outputs and was tested, together with each individual parser, on the test set. As Surdeanu and Manning (2010) note, our approach in combining independent parses of the same sentences can be described, as they put it, as "a meta-classifier that selects candidate dependencies based on their likelihood of belonging to the correct tree." Furthermore, they assert that this strategy did not offer significant gains over the standard, "unweighted" MV but, as far as our experiments on Romanian dependency parsing go, Tables 2 and 3 show that the SSPR meta-classifier is able to improve the LAS of the MV ensemble by 1.1% (from 81.86% to 83%). We can explain this significant increase as follows:

- Surdeanu and Manning use seven parsers in their ensemble but six out of these seven parsers are, in fact, the same parser (the MALT parser) with different linking strategies. We postulate that these six flavours of the MALT parser output very similar parse trees such that the case where three or less parsers (the minority subset) agree on the correct dependency is poorly represented;
- We use three different parsers, each with its own modelling of the parsing problem, such that the case where only one out of the three parsers (the minority subset) finds the correct dependency relation adds a *significant* 6.8% to the LAS of the MV ensemble. It is this pool of correctly found dependency relations from which SSPR is able to extract the 1.1% improvement of the MV ensemble LAS;
- Lastly, we think that our NN-based meta-classifier is superior to the L2-regularized logistic classifier that Surdeanu and Manning used to combine parses, mainly because we rely on a richer set of features that also include semantic features and second-order features.

Table 3 shows that the SSPR NN ensemble parser is 1.6% LAS points and 1.5% LASpr points better than the best individual parser **(**MATE**)**. The medium LASpr score of 84.61% places SSPR in the same class with the best parsers in CoNLL-2017 task for Romanian (see Figure 2). Even though our evaluation setting is not identical to the one for CONLL 2017, the working corpus is the same and our 10-fold cross validation strategy provides a more robust result, *for Romanian*, than the CONLL 2017 1-fold run (the CONLL final parser ranking is backed up by the runs in various languages).

Figure 2: LASpr scores in CONLL 2017 parsing shared task

The LAS medium score of 83% reflects the general performance of SSPR, regardless of its domain of application, because the treebank we worked on is balanced and contains texts from various domains. Table 4 shows the variation of LAS according to the domain. It can be noted that juridical, medical and scientific (from mathematics, physics and computer science) texts have over the medium scores: in their case, it is relatively simple to identify the elements between which a relation is very probable and this relation type is easy to predict (or machine learn): the topic is quite stable, ellipses are rare (a consequence of the need for clarity), predicates saturate their valences locally by words from predictable semantic classes. At the opposite pole, the literature domain, even if it does not contain long sentences, raises the most parsing problems: the authors' creativity manifests both in the sentence structure (dislocated arguments, unlexicalised ones, unusual word order, etc.) and in unusual combination of words (with spectacular

stylistic results), namely words from unexpected semantic classes taken as arguments.

SSPR	Average
Belletrist	**0.7839**
Juridical	**0.8694**
Medical	**0.8571**
Academic	**0.8332**
Encyclopaedic	**0.8001**
Journalistic	**0.8373**
Scientific	**0.8812**
Miscellanea	**0.8053**

Table 4: LAS for SSPR according to the domain

Conclusions

We presented evidence that Romanian dependency parsing can be improved by using "better informed" ensembles of individual dependency parsers, in our case in the form of a neural network that learns to distinguish correct links from incorrect ones. The improvement margin can only increase when adding more, *as different as possible*, parsers into the mix that uses our supplementary linguistic features and we plan to extend this work by adding more such parsers to the ensemble. Specifically, we need to see what parsers from the CONLL 2017 shared task are open-sourced and readily available such that new ways of modelling dependency parsing could be of benefit to our ensemble parsing. One such example is the best parser from the CONLL 2017 shared task on Romanian, namely the Stanford parser which is a NN-based dependency parser.

Another direction we could take to improve ensemble parsing (or individual parsing for that matter) would be to better generalize a dependency relation. Our approach of using semantic features worked to a certain degree but, since NNs work well with "embeddings", we can think of constructing "dependency relation embeddings" as real-valued vectors that are computed with recursive NNs.

Bibliografical References

Barbu Mititelu, V., Ion, R., Simionescu, R., Scutelnicu, A., Irimia E. (2016a) Improving parsing using morpho-syntactic and semantic information, in *Revista Romana de Interactiune Om-Calculator 9(4)*, 285-304, 2016.

Barbu Mititelu, V., Ion, R., Simionescu, R., Irimia, E., Perez C.-A. (2016b) The Romanian Treebank Annotated According to Universal Dependencies., Proceedings of The Tenth International Conference on Natural Language Processing (HrTAL2016), Dubrovnik, Croatia, 29 September – 1 October 2016

Bohnet, B. (2010). Very High Accuracy and Fast Dependency Parsing is not a Contradiction. The 23rd International Conference on Computational Linguistics (COLING 2010), Beijing, China.

Boroș T. and Dumitrescu, S.D., Multilingual tokenization and part-of-speech tagging. Lightweight versus heavyweight algorithms, to be published in Lecture Notes in Artificial Intelligence, 2018.

Călăcean, M., Nivre, J. (2009) A Data-Driven Dependency Parser for Romanian, Proceedings of the Seventh International Workshop on Treebanks and Linguistic Theories, 65-76.

Chu, Y. J.; Liu, T. H. (1965). "On the Shortest Arborescence of a Directed Graph", *Science Sinica*, 14: 1396–1400

Colhon, M., Cristea, D. (2016). Dependency Parsing within Noun Phrases with Pattern-based Approaches, Proc. of the 12th Int. Conf. Linguistic Resources and Tools for Processing the Romanian Language, 51-60.

de Marneffe, M.-C., Dozat, T., Silveira, N., Haverinen, K., Ginter, F., Nivre, J., Manning, Ch. D. (2014). Universal Stanford Dependencies: A cross-linguistic typology. In *Proceedings of LREC*, 4585-4592.

Dumitrescu, S. D., Boroș T., Tufiș, D. (2017). RACAI's Natural Language Processing pipeline for Universal Dependencies, Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, August, 2017, Vancouver, Canada, Association for Computational Linguistics, 174-181.

Edmonds, J. (1967), "Optimum Branchings", *J. Res. Nat. Bur. Standards*, 71B: 233–240.

Hristea, F.,Popescu, M. (2003) A Dependency Grammar Approach to Syntactic Analysis with Special Reference to Romanian. In F. Hristea and M. Popescu (eds.), *Building Awareness in Language Technology*, Bucharest, University of Bucharest Publishing House, 9-16.

Ion, R. and Ștefănescu, D. Unsupervised Word Sense Disambiguation with Lexical Chains and Graph-based Context Formalization. In Proceedings of the 4th Language and Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics (Vetulani, Zygmunt). November 2009, Poznan, Poland, pp. 190-194,

Ion, Radu. (2007). Word sense disambiguation methods applied to English and Romanian, PhD Thesis, Romanian Academy.

Lei, T., Xin, Y., Zhang, Y., Barzilay R. and Jaakkola, T. (2014) Low-Rank Tensors for Scoring Dependency Structures. ACL 2014.

Nivre, J., J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov and E. Marsi (2007). MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2), 95-135.

Perez, A.C., Mărănduc, C., Simionescu, R. (2016) Including Social Media – A Very Dynamic Style – in the Corpora for Processing Romanian Language. In: Trandabăț D., Gîfu D. (eds) Linguistic Linked Open Data. RUMOUR 2015. *Communications in Computer and Information Science*, vol 588. Springer, Cham, 139-153.

Seretan, V., Wehrli, E., Nerima, L., Soare, G. (2010) FipsRomanian: Towards a Romanian Version of the Fips Syntactic Parser, Proceedings of LREC 2010, 1972-1977.

Surdeanu, M. and Manning, C. D. (2010). Ensemble Models for Dependency Parsing: Cheap and Good? In Proceedings of the HLT 2010 Human Language Technologies: The 2010 Annual Conference of the

North American Chapter of the Association for Computational Linguistics, 649-652.

Tufiș, D., Barbu Mititelu, V., Irimia, E., Dumitrescu, S.D., Boroș, T. (2016) The IPR-cleared Corpus of Contemporary Written and Spoken Romanian Language. In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016). Portorož, Slovenia, pp. 2516-2521, May 2016

Tufiş, D., Barbu Mititelu, V. (2014) The Lexical Ontology for Romanian, in Nuria Gala, Reinhard Rapp, Nuria Bel-Enguix (Ed.), *Language Production, Cognition, and the Lexicon*, series Text, Speech and Language Technology, vol. 48, Springer, p. 491-504