Two Level Approach to SRL

Luca Dini, Milen Kouylekov, Marcella Testa, Marco Trevisan
CELI S.r.l., Torino Italy
{dini,kouylekov,testa,trevisan}@celi.it

Abstract. In this paper we present CELI's participation in Evalita 2011 FLaIT task. Based on Markov model reasoning, our system obtained the highest precision in comparison to the other participants.

Keywords: Markov Model, Semantic Roles Labeling, FLaIT, Evalita

1 Introduction

The Frame Labeling over Italian Texts (FLaIT) task is an SRL evaluation exercise part of the Evalita 2011 campaign[1]. The task goal was to detect the semantic frames and roles explicitly stated in an Italian sentence according to the Frame Semantics framework defined by [2]. The task was separated in 3 sub-tasks: 1) Frame Labeling; 2) Semantic Roles Boundary Recognition; and 3) Semantic Role Labeling.

In our participation in FLaIT task we concentrated our efforts in developing a Semantic Role Labeling module based on dependency parser and Markov model reasoning. We have optimized the performance of this module on precision and obtained excellent results. We have also developed a complementary components for Frame Labeling based on context similarity and Boundary Recognition based on phrase recognition algorithm from dependency parser output [3].

2 Semantic Role Labeling

Our approach to SRL is based on two assumptions:

1. We must be able to perform SRL in a "real" condition, i.e. having as input a text, not a set of pre-processed lines.
2. We want maximize precision over recall.

The former assumption implies that, given the Evalita corpus for SRL, we always linearize input sentences, we parse them with our dependency parser (cf. Testa et al. 2009) and we apply either our learning procedure (for training) or learned rules (for testing). The latter assumption (centrality of precision) calls for the application of methodologies which can be relatively easily controlled (i.e. influenced) by an expert.

[1] http://www.evalita.it

Our approach try to build on the basic assumption that roles can be assigned mostly on the basis of the subcategorization list, as interpreted in several linguistic theories (e.g. HPSG). In our approach subcategorization lists are deduced from the gold standard as parsed by a dependency parser. Thus the first processing phase has only the goal of deducing frame assignment predicates such as:

sembrare ("VERB^OBJ" "Inference") sembrare ("NOUN^SUBJ" "Phenomenon")

which are interpreted as "the verb *sembrare* can assign the role Inference to a verb in object position", etc. Under a traditional, “principled” approach, one should then just apply these assignment rules to the output of the parser to obtain the semantic labeling of the arguments. Unfortunately such a "clean" approach is hampered by a set of factors, such as:

1. Parsing error;
2. Verbal alternation which is able to shift argument assignment (unaccusativity, diathesis etc.)
3. Ambiguity of verbs with respect to argument assignment
4. Ambiguity of argument themselves (e.g. a PP with the same head which can receive two roles).

Therefore we mixed the pure subcategorization assumption with a more machine-learning based approach. However, in order to keep both the statement of the importance of principled assignment and to cope with overfitting, we adopt an approach based on Markov Logic Networks [5] and in particular we borrow both the methodology and the software from [4] (code: thebeast, http://code.google.com/p/thebeast/). Basically Markov Logic is a statistical Relational Learning language based on First Order Logic and Markov Networks. It allows writing defeasible first order predicates which are evaluated against a set of facts (worlds) in order to deduce if an unseen predicate holds true. Predicates are normally weighted and most systems expose algorithms which allow the automatic computation of weights on the basis of a gold standard of true worlds. In particular the approach followed by Riedel (2008) to deal with SRL consists in allowing the user to write *template* rules which are then expanded in all logical possibilities and then assigned a weight on the basis of the gold standard. For instance a (simplified) rule such as:

for Int a, Int p, Lemma l, Role r, FrameLabel fr **if** plemma(a,l) & isPredicate(p) & possibleArgument(a)& evoke(p,fr) **add**[role(p,a,r)] * w_lemma_sframe_a(l,r,fr);

is interpreted as "In a sentence (world, in MLN terms), given a certain predicate which evokes a certain frame and a possible argument with a certain lemma assign a certain role with a certain probability (the value of w_lemma_sframe_a)". The system takes care of instantiating this rule with all possible values for lemma and frame and verify on the gold standard which values are meaningful, and which score/weight should be assigned to the occurrence of a certain tuple of predicates satisfying the rule. The biggest part of the configuration work then consists in designing appropriate rules, which maximize precision without degrading too much the performance (MLN

are notorious for being extremely inefficient in the weight learning phase). In our case we adopted the following classes of rules:

1. linear rules considering word features (POS) in a certain window (5)
2. rules considering distance between the predicate and its possible arguments.
3. rules taking into account the compatibility of certain features of the predicate word and its possible argument word (lemma, surface form, part of speech, frame…)
4. rules considering dependency relations between the predicate and its possible arguments.
5. rules taking into account the computed subcategorization list and the features of the possible argument.

In order to increase a little bit the recall, the latter class of rules is expanded by relaxing certain constraints (e.g. subcategorization assigned to a lemma can be expanded to subcategorization assigned to a frame, valence based on grammatical function and part of speech tag can be expanded as a disjunction of the two, etc.). Still our results were somehow deceiving with respect to the ones mentioned in Riedel 2008[2] as our internal evaluation reported Recall=0,455, Precision=0,703 and F1=0,553[3] . After manual inspection of the errors, focused to explain the low recall, we noticed that some apparently obvious cases of role assignment based on the sub-categorization list were missing and that some role was wrongly assigned simply on the basis of word combination rather than dependency (this can be probably imputed to the relatively small dimensions of the corpus, which might risk to privilege idiosyncratic rules)

A second layer of rules was therefore added to "correct" the output of the trained theBeast system. As these rules are manually coded (no template expansion) and as we wanted to have weights assigned by humans, we made use of another MLN implementation i.e. Tuffy [1] (http://research.cs.wisc.edu/hazy/tuffy/).[4] The new rules are all dependency based and look like the following:

11 !evoke(v0, v1, Statement_proclamare) v !dep(v0, v1, v2, SUBJ) v assignT(v0, v1, v2, Speaker)

-2 !evoke(v0, v1, Statement_dire) v !dep(v0, v1, v2, OBJ) v assignT(v0, v1, v2, Occasion)

These are interpreted as "if an instance of the verb *proclamare* with frame Statement has a subject, then the subject is likely to be a Speaker" and "if an instance of the verb *dire* with frame Statement has an object, then the object is unlikely to be a

---

[2] Possible causes are: 1)the dimension of the weight learning corpus; 2)the fact that Reidel's experiment was based on manual annotation of dependencies, pos and syntax; 3)the fact that more features were available in his experiments.

[3] The figures are slightly different from the official evaluation figures: this is probably due to the fact that: 1) they are based on a portion of the gold standard (not used for rule weighting) and not on the test set; 2)the evaluation software (we used the mechanism internal to the theBeast package) consider role assignment to a lexical head, not to the phrase

[4] The reason why we adopted this implementation is that we found tuffy performs better than tuffy in assigning global and hard constraints

Speaker". Other classes of rules are more generic and are mainly meant to increase recall:

13 !evokeFrame(v0, v1, GivingFr) v !dep(v0, v1, v2, OBJ) v assignT(v0, v1, v2, Theme)

which reads as "if a verb evokes the frame Statement and has a direct object, then the direct object is likely to be a Theme".

The new set of rules apply on top of theBeast assignments, which means that all initial features *and* theBeast assignments are visible to them. The final assignment is then determined by a set of Tuffy meta-rules (i.e. rules taking into account both Tuffy and theBeast output)

3 Frame Labeling & Boundary Detection

The frame labeling and boundary detection module were developed for our participation.

3.1 Frame Labeling

For Frame Labeling we employed a similarity based approach. The core of the approach assigned for each candidate words the frame that has more similar words examples in the training set. We define similarity between two words as the cosine similarity between the words in the immediate context of the two words. **Cosine similarity** is a measure of similarity between two vectors by measuring the cosine of the angle between them. The cosine of 0 is 1, and less than 1 for any other angle. The cosine of the angle between two vectors thus determines whether two vectors are pointing in roughly the same direction. To generalize the approach we did not compare the context of each candidate word to the context of each annotated word in the Training Set but to a generalized set of words for each frame, that were encountered frequently in the contexts of the words annotated with this **frame**. For example the Frame **Cause_harm** had the following words: "s*fruttare, torturare, gravemente, minacciare and incident"* as part of its generalized set.

3.2 Boundary Detection

Boundary detection has been performed using specific grammatical rules encoded by our dependency parser, described in [3].

LFC parser, that works at different levels (disambiguation, chunking and dependency). The dependency module uses rules to identify syntactic dependencies between linguistic units (or nodes in the chunk tree). It should be noticed that the grammar computes dependencies holding semantic heads, rather than syntactic.

We began with core grammar relations (i.e. dependencies relations, such as subject or object) and constructed upon them to create more specific and complex dependencies, each of which representing a simple phrasal constituent. Each dependency has been built in order to be a relation of 3 arguments: *label* (*arg1, arg2, arg3*). Given a certain syntactical pattern, the label is the name of the dependency; arg1 is a token that represents the head of the dependency, while arg2 means the left boundary and arg3 the right one. Below is an simple example of a completive sentence recognition:

Sentence: *A favore delle popolazioni di regioni colpite da catastrofi.*
Relation:PREPOSITION_PHRASE(regioni, di, colpite)
PREPOSITION_PHRASE(popolazioni,a favore delle, popolazioni)

We have used the grammatical relations identified in the sentence as a potential boundaries of the roles. For each role head we selected as boundary the longest phrase that had the role head as first argument or contained the it as part of the phrase.

4 Results

Table 1. Results Obtained

	Run	FL	BR	BR Token	AC (P)	AC (R)	AC(F)	AC Token (P)	AC Token (R)	AC Token (F)
WT	1	69.23	30.27	40.58	27.41	16.25	20.40	49.49	20.93	29.42
NT	1	69.23	28.71	46.77	32.55	14.82	20.37	47.90	15.98	23.96
WT	2	X	32.56	40.89	31.23	19..46	23.98	64.56	27.49	38.56
NT	2	X	30.13	47.92	36.12	16.96	23.09	62.26	20.50	30.85
WT	3	X	X	X	**75.0**	40.18	52.33	**83.24**	51.49	**63.62**
NT	3	X	X	X	73.23	32.32	44.86	76.58	36.34	49.29

We have submitted two systems for evaluation the first one had manually encoded rules activated and the second one didn't.

The frame labeling results were a disappointment to us. The poor performance can be explained by the fact that according to our observations there were in the test set *20* sentences containing a predicate word for which no frame annotation was found as an example in the training set.

Our improvised approach to Boundary Recognition is the place where our system can be improved significantly.

The results obtained clearly demonstrate the impact of manually encoded rules. Improving the Frame Labeling and Boundary Recognition tasks in the second and third run the system using it clearly outperforms the one without.

We were satisfied with the precision obtained by the system on the Argument Classification task which outperform the other systems reaching 75.0 F-measure.

5 Conclusions

Here we presented a Markov Logic Network based approach to Semantic Role Labeling which tries to maximize on linguistic features obtained by a dependency parsing. The approach focuses on "pure" semantic labeling whereas word sense disambiguation and phrase boundary detection are considered ancillary tasks (the whole approach has been designed to assign role to lexical heads). We think that globally two conclusions can be drawn from this experience:

1. The mixed approach with learnt weights and manually coded rules seems promising: indeed the addition of manual rules over learn weight increases F1 by 6% (AC) and 14% (ACTB). It would be interesting to check whether improvements are possible even on top of systems whose first layer in not based on MLN.
2. The approach is heavily dependent on the quality of the dependency parser. Given the centrality which is assigned to subcategorization lists, as a matter of fact role assignment takes place in most cases *only* when a dependency exists between the frame bearing element and the lexical head to which the role should be assigned. This constitutes an explanation of the poor recall of our system as compared to the high precision.

References

1. Niu, F., Ré, C., Doan, A., Shavlik, J.W.: Tuffy: Scaling up Statistical Inference in Markov Logic Networks using an RDBMS CoRR abs/1104.3216 (2011)
2. Fillmore, C.J.: Frames and the semantics of understanding. Quaderni di semantica, 6(2):222-254 (1985)
3. Testa, M., Bolioli, A., Dini, L., Mazzini, G.: Evaluation of a Semantically Oriented Dependency Grammar for Italian at EVALITA. In: Proceedings of EVALITA 2009 (2009)
4. Riedel, S.: Improving the accuracy and Efficiency of MAP Inference for Markov Logic. In: Proceedings of the 24th Annual Conference on Uncertainty in AI (UAI '08) (2008)
5. Richardson, M., Domingos, P.: Markov Logic Networks. Machine Learning, 62, pp 107--136 (2006)