Semantic Role Labeling : A short history

Utkarsh Upadhyay

November 9, 2008

Abstract

This term paper talks about the short history and recent works in Semantic Role labeling, beginning from the seminal paper by Daniel and Jurafsky [5] then onto two recent publications, which talk about robustness of existing parsers [8], and the importance of parsing in semantic role labeling [9]. This paper, hence, in short discusses the development of Semantic Role Labeling, so far and the state of the art in the field at this point in time.

1 Introduction

The common thread in all the three papers is Semantic Role Labeling. Hence, it will be described in a little detail in the following section. Following it would be the reason why it is important and the reason for increase in interest in the field [1]. The seminal paper of Gildea and Jurafsky [5] would be discussed next. The contributions of the other two papers would be discussed next.

2 Semantic Role Labeling

Semantic Role Labeling is segmentation and classification of a sentence into certain arguments, such that they can be used to answer general questions. Initially it was thought that hand-crafted domain-specific grammar rules would be able to do the task. These systems were built and were successfully deployed to perform simple tasks like report of bank balances or answer queries regarding flight times, etc. For this task, careful hand-crafted rules and templates were made. The templates for such a system had entries like:

- ORIG_CITY
- DEST_CITY
- DEPART_TIME
- ...

while a similar system intended for analysing and understanding mergers and acquisitions would have the frames:

- PRODUCTS
- RELATIONSHIP
- JOINT_VENTURE_COMPANY
- ...

There is a clear possibility of making these chunks more general so that they apply to a larger range of sentences. Hence, while working on FrameNet (1998) [2], the sentence frames as described in Fillmore [6], were used. These frames described a generic classes where a sentence can belong semantically, e.g. TRANSFER frame, CONVERSATION frame, JUDGEMENT frame, etc. Such 12 total frames were recognized. Also, they are further divided into various participants (e.g. SPEAKER, MESSAGE, and ADDRESSEE for the SPEAKER frame; JUDGE, EVALUEE, and REASON to the JUDGEMENT frame, etc.) making a total of 67 such sub-categories.

Upon observing these categories made in such a way, it can be seen that this lies at a point between dividing the language into completely domain specific roles and reducing it to the KARAK-KARTA theory, or to PROTO-AGENTS and PROTO-PATIENTS. However, making a compromise, we can settle for such a representation. Also, adopting this kind of a representation, the FrameNet has already been hand annotated. Later, PropNet was built on similar principals, but with lessons learnt from the FrameNet, and hence, is not regarded as the *de facto* standard to test Semantic Role Labelers against.

3 The beginning

3.1 The first attempt : Automatic labeling of semantic roles

The first attempt to solve the problem was made by Gildea and Jurafsky in 2002, while the FrameNet itself was in a preliminary stage, containing only 50,000 sentences. Also, FrameNet was not very well parsed,

the meaning and exact effect of which we will soon see. Also, their performance on the sentences was not stellar. They, nevertheless, were able to spark immediate and intense research in the area, primarily by introducing a novel set of features of text, which were extensively used in almost all subsequent research, and an initial structure of the solution, which also has so far been the most popularly used and the most successful architecture.

3.1.1 The two stages architecture

The architecture used by Gildea and Jurafsky was two layered, with the identification the argument boundaries being treated as a separate problem and the assignation of a class to the segment as a different problem. As parsed data was already available with FrameNet, they were able to test their Identification and Classification tasks separately.

3.1.2 The test features used

Most of their features were derived from a parse tree of the sentence. This subsumes the presence of a parser and its correctness, an assumption that was put to test by Punyakanok, Roth and Yih [9]. The Collins parser was used for parsing as it was among the best known and the most robust parser available at that time.

1. **Phrase Type** : of the constituent. For example in a communication phrase:
   - **SPEAKER** is generally NP
   - **TOPIC** is generally PP
   - **MEDIUM** is generally PP
2. **Governing Category** : *Only for NPs* Whether the phrase is governed by a S or by a VP, that is which node we come across first while ascending the parse tree from the given constituent.
3. **Parse Tree Path** : Path in the parse tree from the target word invoking that frame to the constituent forming a part of that frame. The following table shows the strong correlation between the semantic property of the constituent and the path.

Freq.	Path	Description
14.2%	$VB \uparrow VP \downarrow PP$	adjunct
11.8%	$VB \uparrow VP \uparrow S \downarrow NP$	Subject
10.1%	$VB \uparrow VP \uparrow NP$	Object
...		

4. **Position** : Whether the argument is present before the predicate invoking that frame or after it. This also, overcomes errors due to incorrect parses.
5. **Voice** : Active / Passive. This helps in deciding whether we expect the subject/object to be present before or after the predicate.
6. **Head Word** : Head words from the constituent phrases taken as according to Collin (1999) [3]
7. **Subcategorization** : *Only for Verbs*: The structure of the node just above the verb in the parse of the sentence. For example:
   (a) *He opened the door.* : **opened** has subcategory of $VP \rightarrow VB - NP$
   (b) *The door opened.* : **opened** has sub-category of $VP \rightarrow VB$
8. **Frame Element Group** : *Only for Verbs* : The possible frames that might be present with a verb, and their probabilities. This was available to them if they assumed that the FrameNet was exhaustive, and then performed a search for each predicate over it.

3.1.3 Probability Estimation

They attempted various methods for performing the final classification, including Linear Interpolation of probabilities for unknown points in the data (unprecedented tuples of data, e.g. *(**Head Word**=fall, **Voice**=Passive, **Position**= After)* may not have occurred in the data anywhere. Finally, however, they settled on a back-off probability prediction model.

They used a Lattice based structure to predict the probability of each constituent belonging to one category. The data was very sparse owing to large number of categories and limited number of sentences available to them. Hence, there were various features tuples that had *never* been observed in the entire FrameNet. To work around this problem, they used their best estimators (The probability estimators which were the most accurate) at the top of the Lattice structure and used it for classification. However, if the data for the top most layer wasn't present, then they descended down the lattice structure and tested for highest probability there.

3.1.4 Performance

They obtained commendable performance with their system, which was marred by various factors such as sparse training data, erroneously marked data and no-precedents. These difficulties were impressively overruled and the final numbers were:

- 82% accuracy in identifying pre-segmented data.
- While simultaneously identifying segments and labeling them: 65% precision and 61% recall.

4 Development

The development of Semantic Role Labeling burgeoned into the academic scene with a flurry in the subsequent years, and the following list will give a taste of the exciting influx of ideas:

- **2002**: *Automatic Labeling of Semantic Roles* by David Gildea, Daniel Jurafsky: In which paper the possibility of Automated Semantic role labeling was first discussed.
- **2003**: *Semantic Role Labeling by Tagging Syntactic Chunks* by Kadri Hacioglu, Sameer Pradhan, Wayne Ward, James H. Martin, Daniel Jurafsky : This was the first *break through* that SRL needed, as this was the first paper that saw SRL as a massive classification task and attempted to use multiple SVMs for the puspose. [7].
- **2004**: CoNLL[1] '04, which is a reputed conference, had Semantic Role Labeling as one of its shared tasks that it used to have every year.
- **2004**: *Pruning heuristics for Two Stage SRL systems* by Xue and Palmer: This was the first paper where the author realised the importance of global information present around the sentence we are parsing and tried to incorporate that information using an Inference engine to prune the search space in the classes field further still.
- **2004**: *Semantic role labeling via integer linear programming inference* by V Punyakanok, D Roth, W Yih, D Zimak: Instead of meagerly pruning the tree, they engaged a linear programming approach and then were able to perform Inferences that led to massive pruning of the trees.
- **2005**: ***PropBank completed.*** : It marked the era of a new kind of classifiers now that the number of classes of arguments were further abstracted out and given a mere 17 categories. Also, care was taken so that PropBank's arguments were complete syntactic units (NPs, VPs), thereby reducing the errors further by about 13%, which were the estimated number of mistakes made in FrameNet hand annotations.
- **2005**: In CoNLL '05, shared task was yet again SRL, which again resulted in an array of useful research projects in the field.
- **2005**: *Semantic Role Labeling: A sequence tagging problem* by Marquez, Pere Comas, and Catala: The technique used in this paper is not based on the two layer architecture that we had discussed. It instead treats the input words as a sequence of input symbols and tries to predict whether this is an extention of existing arguments or is a new argument itself.

Hence, it is clear to see that this field of study has very much been in picture as a nascent fertile research topic since its birth in 2002.

[1]Conference on Natural Language Leaning

5 State of the art

5.1 Towards Robust Semantic Role Labeling [8]

As I have already lauded them for, they were the pioneers of attempting to use multiple SVMs on this task of semantic role labeling. They also assumed the presence of a parser for various tasks (the Charniak parser), and in their paper have tested the robustness of this parser over two different sets of data, using different training-testing combinations. The conclusion of their study is that the parser errors are insignificant when compared to the other kind of errors they observe in the classification. Hence, the problem according to them is that semantic role labeling is fundamentally hard for heterogeneous data.

5.1.1 Formulating the problem for SVMs

SVMs have been shown to perform well for classification of data with a high dimensionality [4]. Also, they have been shown to be generally good at classification tasks with relatively little or no tuning, unlike Artificial Neural networks. The model that they have deployed is a one v/s all, in which they have to train as many classifiers as they have classes. Also, since the SVMs only provide us with a distance of the new vector from a hyperplane, a sigmoid function is used to convert the distance into a probability value.

However, it is easy to see that SVMs are not readily applicable to the problem, especially if they work independently on each constituent irrespective of what will be the value of the other constitutes in the sentence. There always is the problem of extraneous assignation happening, like a sentence having two predicates, or of erroneous assignations, like two overlapping nodes being classified as arguments. To avoid this problem, they make a Viterbi search like algorithm to search through the various possibilities and decide the most probable one.

The model made is similar to the Markov models

- **States**: *n-best* hypothesis of the classes as dictated by SVMs.
- **State probabilities**: Tri-grams of possible classes sequences (trained along with the SVMs)

- **Observation Probabilities**: The observation probabilities are obtained from the SVMs, using the sigmoid function.
- **Search**: Is constrained such that no two overlapping nodes are given a NON-NULL label.

5.1.2 Robustness

The robustness of their system ASSERT, was tested by using the following corpus:

1. PropBank Wall Street Journal corpus.
2. PropBank Brown Corpus data.

5.1.3 Performance

The performance of ASSERT overall was the following:

Parse	Task	Prec	Rec	Accr. (%)
TreeBank	Id.	97.5	96.1	-
	Class.	-	-	93.0
	Both	91.8	90.5	-
$Auto^2$	Id.	87.8	84.1	-
	Class.	-	-	92.0
	Both	81.7	78.4	-

However, their primary conclusion was that performance on Brown Corpus was more difficult, as was reflected by numbers, because the problem is fundamentally harder to solve for heterogeneous data. Also, one of their conclusions was also that parsing is ***not*** a problem any more as the parsers performed reasonably well under training from one set and testing on the other data set. Identification's robustness was evinced in the tests: better the parser, the better is the identification of the arguments.

Their first suggestion was that instead of choosing very restrictive features that depend on the text format, one should attempt to work with as general features. As an example, instead of the head words of sentences, one should choose an abstraction over it. This kind of a choice may initially drop the performance of WSJ data on itself, but this will certainly generalise better.

Also, observing the degraded performance on Brown's data, clearly one of the primary conclusions in that more diverse training and testing data should be used for the task now. So far, most research in Semantic Role Labeling has remained closely associated with the WSJ data source. The author's claim is keeping both the corpus together will provide a better training as well as a better testing set.

5.2 Importance of Parsing and Inference in SRL [9]

The authors here primarily discussed the importance of parsing and inference in performing Semantic Role labeling. They were the pioneers in using Linear programming for pruning the features derived from parse trees in the ability of performing an accurate Semantic Role labeling. The authors have taken into account features provided by shallow parsers and have tried to train their system using only those features to compare the performances. Not quite to their surprise, they discover that the parsing features are though not very important while performing classification. However, while identifying the constituents' boundaries, the presence of a full parse tree played a very major role. According to them, the presence of a true parse tree helps greatly in reducing the total search space very much for their linear programming based inference procedure [10].

5.2.1 Multiple parsers

To hedge against parsing errors, they used the best five parses produced by the Charniak parser trained over Penn TreeBank and a parse provided by the Collin's parser. Hence, their labeler was more or less immune to parsing errors.

5.2.2 The shallow parse features

The authors have tried to imitate the features very similar in likeliness to the features obtained from a full parse tree. In shallow parsing we have available with us chunk level information. The new features obtained from shallow parses were:

- Chunk lengths
- Chunk Types $\sim$ *Phrase Type*
- Sequence of chunks from the chunk to the predicate $\sim$ *Tree path*
- ...

5.2.3 The inference structure

Another innovative element in their design was the Inference engine which they used to prune the various possibilities. For this a detailed study of the PennBank was needed. Some of the constrains were almost trivial (e.g. All nodes are assigned one and only one class or NULL), while some of the constrains required some special knowledge of the structure (e.g. An R-ARG or a C-ARG must have an original ARG associated with it). This information is not limited to a mere chunk, and, hence, the structure of the entire sentence is taken into account while attempting to look for a possible fit. The overall structure of their inference engine is given below:

- Constrains were designed (from the most obvious to the less obvious):

1. Arguments cannot overlap
2. If there exists a *R-arg*[3] then a parent *arg* must be present.
3. Given the predicate, some frames are illegal: *stalk* only takes ARG0 or ARG1.
   ...

- A linear program was made for the constrains:
  - p_{ic} is the *score* (a function of probabilities of this class using various parses) of the constituent S^i being of class c^i. u is the classification matrix where u_{ic} is 1 iff the i^{th} constituent has the class c and zero otherwise.
  -
$$u^* = \underset{u\in\{0,1\}^d}{argmax}\ p.u \quad (1)$$
    And the constraint:
$$C_1.u \geq b_1 \quad (2)$$
$$C_2.u = b_2 \quad (3)$$
    With p being treated as a cost vector. This is the standard form of linear programming problems and many engines exist for solving these equations.
  - Then the for the classes:
$$\hat{c}^{1:M} = \underset{c^{1:M}\in C}{argmax} \sum_{i=1}^{M} s(S^i = c^i) \quad (4)$$
$$= \underset{u,c}{argmax} \sum_{i=1}^{M}\sum_{c\in C} p_{ic}u_{ic} \quad (5)$$
$$(6)$$
    where $\hat{c}$ is the desired sequences of classes. So far, if we solve the problem without any constrains, we might get illegal argument allocations. Hence, the solution is subject to a few constrains.
  - *Encoding the constrains*: The constrains are formed as a
$$\sum_{c\in C} u_{ic} = 1 \quad \forall i \in [1, M] \quad (7)$$
    In plain English, the constrain says that each segment gets only one class. Similarly other constrains are formulated and imposed on the solution.

[3]Reference Argument

The authors agree that solving a Integer Linear Program is in general NP-hard, but in practice its competitively fast, and they see it as a trade off between the non-optimality possible in beam search and speed.

5.2.4 Performance & Conclusions

Their conclusion was that a full parse is indispensable for a good recognition of argument boundaries while performing Semantic Role Labeling. However, it is not apparent from the immideate results. The results on the other hand seem to impress upon us that the difference is between the performance on the Gold standard parses and the Automatic parsing in Shallow parsing is the same. However, that is not quite the case, since, as the authors claim, the errors made in the initial stage propagates through to the other stages, and though the easy cases are resolved well using the Shallow parse information only, for the difficult cases, it is unalble to prune out many possible arrangements of arguments. Hence, the later stages with the shallow parsing information have a harder task given to them, and, hence, perform worse than when only a few possibilities are made available to them (Full parse tree's nodes).

	Full	**Parsing**	**Shallow**	**Parsing**
Parse	Prec.	Rec	Prec.	Rec
Gold	86.22	87.40	75.34	74.28[4]
Auto	77.09	75.51	75.48	67.13

Also, joint inference with multiple probability values from various parsers helps greatly in improving performance.

Parse	**Prec.**	**Rec.**	**F**
Charniak-1	75.40	74.13	74.76
Charniak-2	74.21	73.06	73.63
Charniak-3	73.52	72.31	72.91
Charniak-4	74.29	72.92	73.60
Charniak-5	72.57	71.40	71.98
Collins	73.89	70.11	71.95
Joint Inf.	80.05	74.83	77.35

6 Conclusion

Semantic Role Labeling has certainly come a long way, since its birth to this point in time. However, after the CoNLL conference 2005, there hasn't been a significant innovation in the field. The viewing of this problem as a sequence tagging problem offered some promise, but it has failed to deliver. Some obvious extension of the techniques here are:

- Using a bag of classifiers with SVMs to see whether yet better parsing promises any better results.

- Combining the constrains in their linear programming avatar with the SVMs can yield much better performances. After all, those constrains are what the Viterbi search is subject to while attempting to find a valid classification.

However, looking at the fine details being discussed by the papers now seems to point that there are few directions we can look at. Only increasing computing power and increasing Corpus sizes seem to be the ostensible means of improving the performance of the Semantic role labelers.

Hence, what is needed to enthuse the field again is another break through, and it just might be around the corner.

References

[1] Introduction to the conll-2005 shared task: Semantic role labeling.

[2] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The berkeley framenet project. In *in Proceedings of the COLING-ACL*, pages 86–90, 1998.

[3] Michael Collins. Discriminative reranking for natural language parsing. In *Proc. 17th International Conf. on Machine Learning*, pages 175–182. Morgan Kaufmann, San Francisco, CA, 2000.

[4] Corinna Cortes and Vladimir Vapnik. Support vector networks. In *Machine Learning*, pages 273–297, 1995.

[5] Daniel Gildea and Daniel Jurafsky. Automatic labeling of semantic roles. *Computational Linguistics*, 28:245–288, 2002.

[6] Philip J. Hayes, Er G. Hauptmann, Jaime G. Carbonell, and Masaru Tomita. Natural-language understanding. In *Encyclopedia of Artificial Intelligence*, pages 660–677. John Wiley & Sons, 1987.

[7] Sameer Pradhan, Kadri Hacioglu, Wayne Ward, James H. Martin, and Daniel Jurafsky. Semantic role parsing: Adding semantic structure to unstructured text. In *ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining*, page 629, Washington, DC, USA, 2003. IEEE Computer Society.

[8] Sameer S. Pradhan, Wayne Ward, and James H. Martin. Towards robust semantic role labeling. *Comput. Linguist.*, 34(2):289–310, 2008.

[9] Vasin Punyakanok, Dan Roth, and Wen tau Yih. The importance of syntactic parsing and inference in semantic role labeling. *Comput. Linguist.*, 34(2):257–287, 2008.

[10] Vasin Punyakanok, Dan Roth, Wen tau Yih, and Dav Zimak. Semantic role labeling via integer linear programming inference. In *COLING '04: Proceedings of the 20th international conference on Computational Linguistics*, page 1346, Morristown, NJ, USA, 2004. Association for Computational Linguistics.