J. Lang, M. Lapata (2011): Unsupervised Semantic Role Induction via Split-Merge Clustering. Proceedings of ACL

J. Lang, M. Lapata (2011): Unsupervised Semantic Role Induction with Graph Partitioning. Proceedings of EMNLP

Unsupervised Semantic Role Induction

Presentation by Clayton Greenberg

January 27, 2014

0. Contents

PropBank Configuration

- Each sense of a verb (lemma + sense number) has its own list of semantic roles: frameset.
- Different syntactic configurations corresponding to the same frameset may have their own rolesets.

Role labels in PropBank

- 38 page annotation guidelines document contains instructions for how to label semantic roles in PropBank.
- They have the goal of making labels Arg0 (proto-agent) and Arg1 (proto-patient) consistent across framesets.
- Other labels might have specific meanings, such as ArgA (inducer of action) and ArgM (argument adjunct) but they are not consistent across framesets.

Frameset example

Frameset **edge.01** "move slightly"

Arg0: causer of motion
Arg1: thing in motion
Arg2: distance moved
Arg3: start point
Arg4: end point
Arg5: direction

Ex: [$_{Arg0}$ Revenue] *edged* [$_{Arg5}$ up] [$_{Arg2\text{-}EXT}$ 3.4%] [$_{Arg4}$ to $904 million [$_{Arg3}$ from $874 million] [$_{ArgM\text{-}TMP}$ in last year's third quarter]. (wsj_1210)

Applications of SRL (1/2)

- Information extraction, question-answering, and summarization: provides information that is instrumental for representing shallow meaning outside of syntactic configuration.

$[\text{Joe}]_{A0}$ **broke** the $[\text{window}]_{A1}$ with a $[\text{rock}]_{A2}$.
The $[\text{rock}]_{A2}$ **broke** the $[\text{window}]_{A1}$.
The $[\text{window}]_{A1}$ **broke**.

Applications of SRL (2/2)

- Machine translation: once you have the shallow meaning, you can map back to the syntactic configuration in the target language.

`like`(x,y) ➔ `gustar`(y,x)

Supervised data bottleneck

- Supervised data are not only expensive to produce, *but also in short supply*.
- Previous successes have be domain-specific due to their training data.
- The output of an unsupervised system must be further processed (most likely by humans) to be included in a knowledge base such as PropBank.

Approach typology

Name of semantic role labeling approach:	Corpus is manually tagged with semantic role labels	System uses a role inventory
Supervised Das et al. (2010)	✓	✓
Knowledge-based Swier & Stevenson (2004)	✗	✓
Unsupervised Lang and Lapata (2011)	✗	✗

FrameNet: verbs with similar sets of roles are grouped together (challenging).
PropBank: each verb has its own set of roles.
CoNLL 2008: a gold corpus with annotations adhering to PropBank guidelines.

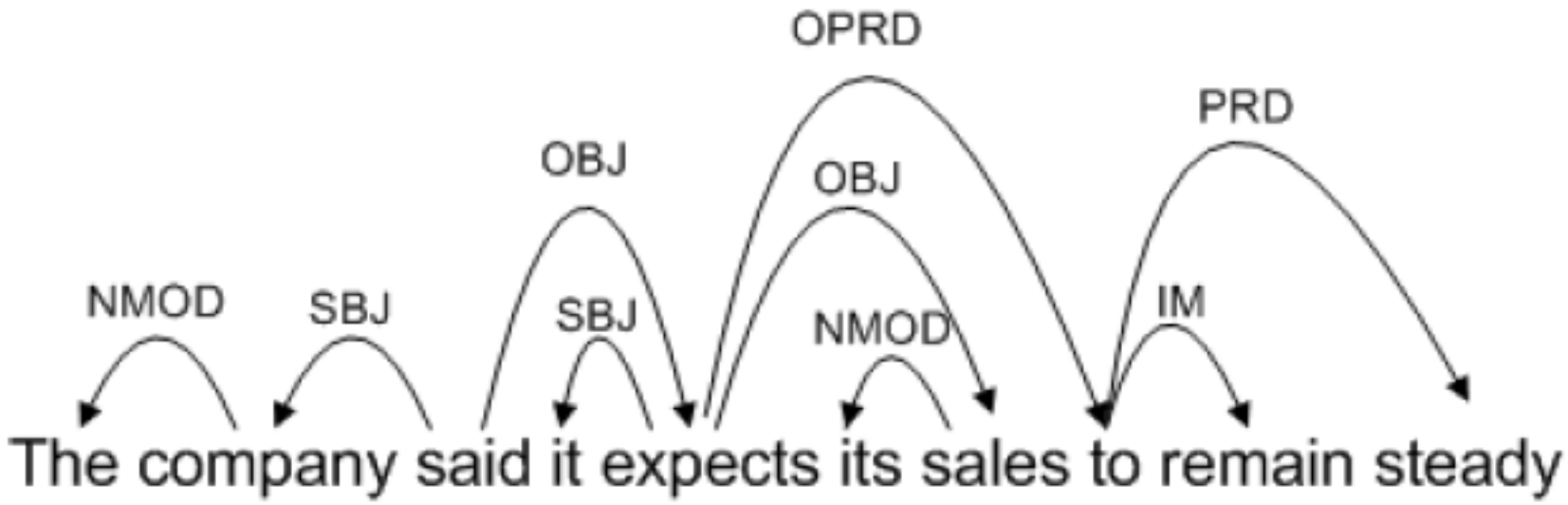
Unsupervised semantic role labeling

- Start with parsed (syntax only) data.
  CoNLL 2008 is dependency-parsed, manually and by MaltParser.
- Argument Identification: determine which words in the sentence are arguments.
- Argument Labeling (Role Induction): cluster identified arguments (lemmata).

Argument Identification

1. Discard a candidate if it is a determiner, infinitival marker, coordinating conjunction, or punctuation.
2. Discard a candidate if the path of relations from predicate to candidate ends with coordination, subordination, etc. (see the Appendix for the full list of relations).
3. Keep a candidate if it is the closest subject (governed by the subject-relation) to the left of a predicate and the relations from predicate p to the governor g of the candidate are all upward-leading (directed as g ➔ p).
4. Discard a candidate if the path between the predicate and the candidate, excluding the last relation, contains a subject relation, adjectival modifier relation, etc. (see the Appendix for the full list of relations).
5. Discard a candidate if it is an auxiliary verb.
6. Keep a candidate if the predicate is its parent.
7. Keep a candidate if the path from predicate to candidate leads along several verbal nodes (verb chain) and ends with arbitrary relation.
8. Discard all remaining candidates.

Argument ID: example



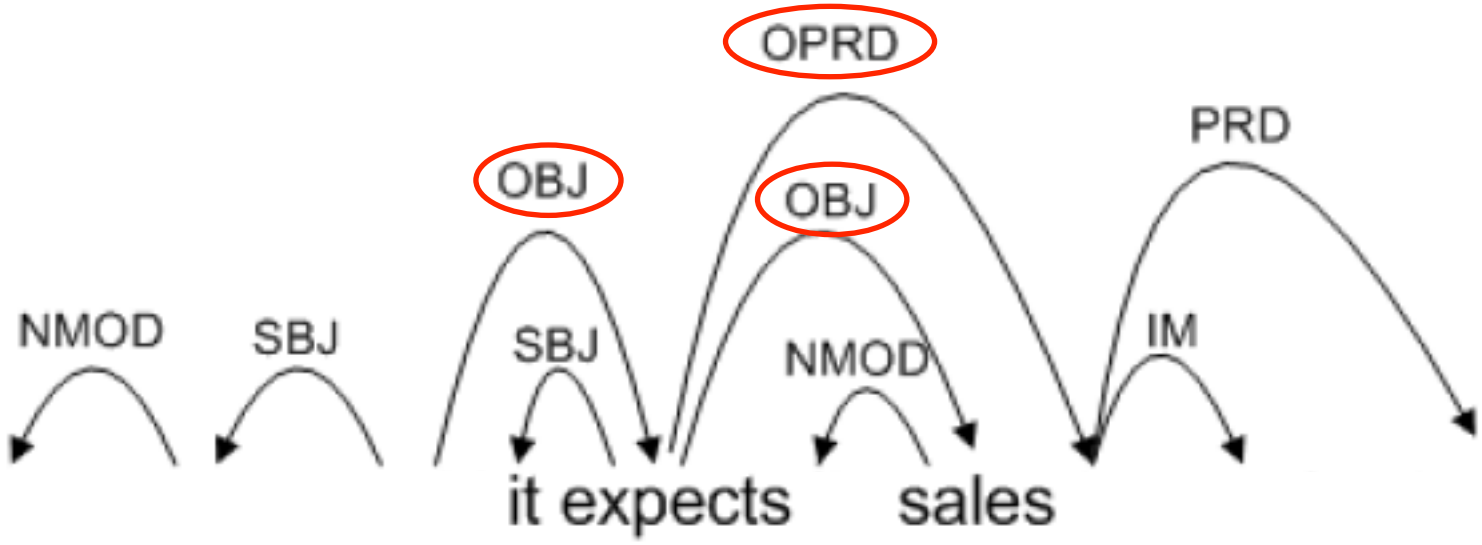
Argument ID: example

Rule 1: Delete determiners and infinitival markers

Argument ID: example

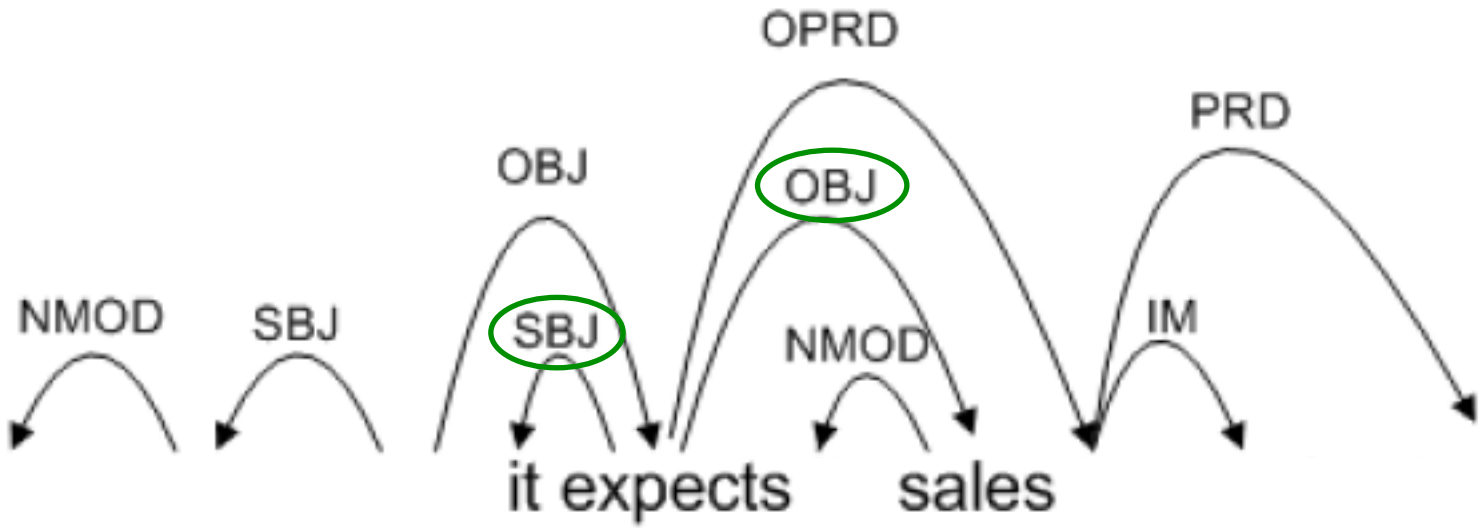
Rule 2: Delete infinitive verb and head standing in object relation to predicate.

Argument ID: example



Rule 4: Delete heads or dependents of words beyond the circled relations.

Argument ID: example



Rule 6: Keep direct dependents of predicate.

Rule 8: Discard everything else.

Split phase

- (Split-merge method) Put all identified arguments into one cluster. Then split based on:
  - verb voice (active/passive)
  - argument linear position relative to predicate (left/right)
  - syntactic relation of argument to its governor
  - preposition used for argument realization
- This gives a high purity (90%) but keeps the clusters from getting too small to be successfully merged by the chosen merge algorithm.

- (Graph-partitioning method) This method does not have the above problem, so initialize each identified argument in its own cluster.

Similarity Criteria

- Word relatedness – whether the arguments found in the two clusters are lexically similar
- Local uniqueness – whether clause-level constraints are satisfied, specifically the constraint that all arguments of a particular clause have different semantic roles, i.e., are assigned to different clusters
- Syntactic configuration – whether the arguments present in the two clusters have similar parts of speech

Split-merge method

Algorithm 1: Cluster merging procedure. Operation $merge(L_i, L_j)$ merges cluster L_i into cluster L_j and removes L_i from the list L.

```
1  while not done do
2      L ← a list of all clusters sorted by number
       of instances in descending order
3      i ← 1
4      while i < length(L) do
5          j ← arg max_{0≤j'<i} score(L_i, L_j')
6          if score(L_i, L_j) ≥ α then
7              merge(L_i, L_j)
8          end
9          else
10             i ← i + 1
11         end
12     end
13     adjust thresholds
14 end
```

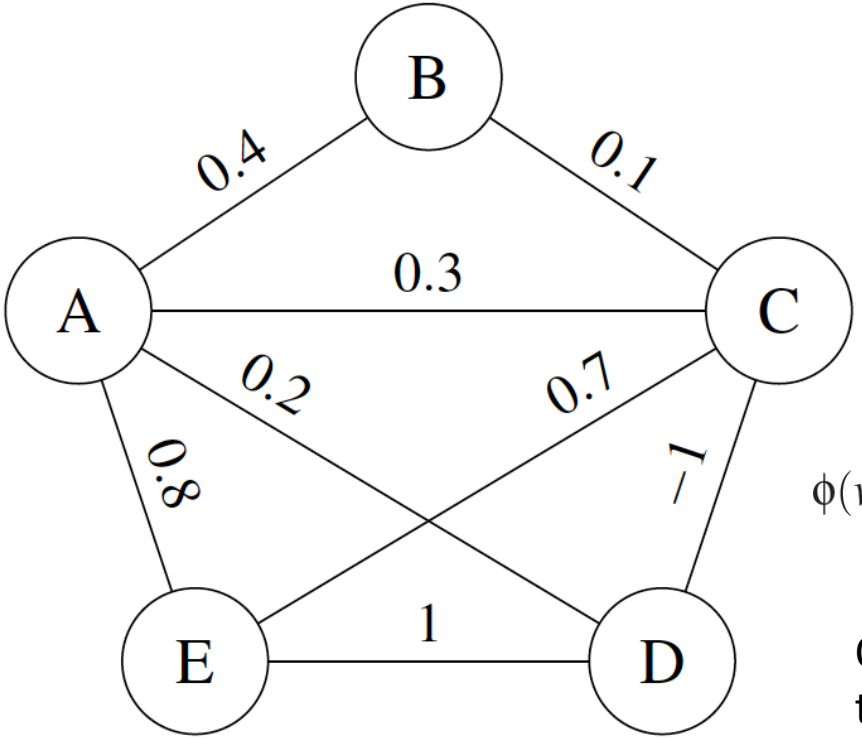
$$score(c, c') = \begin{cases} 0 & \text{if } pos(c, c') < \beta, \\ 0 & \text{if } cons(c, c') < \gamma, \\ lex(c, c') & \text{otherwise.} \end{cases}$$

$$lex(x, y) = cossim(x, y) = \frac{x \cdot y}{\|x\| \|y\|}$$

$$cons(c, c') = 1 - \frac{2 * viol(c, c')}{NC + NC'}$$

Lower β from 0.95 to 0.00 in 0.05 increments.
Lower γ similarly every time β reaches 0.
Then reset β.
Terminate when γ = 0.

Graph-partitioning method



$$l_i \leftarrow \arg\max_{l \in \{1 \ldots L\}} \sum_{v_j \in \mathcal{N}_i(l)} \phi(v_i, v_j)$$

$$conf(l_i \leftarrow l) = \frac{1}{|\mathcal{N}_i(l)|} \sum_{v_j \in \mathcal{N}_i(l)} \phi(v_i, v_j)$$

$$\phi(v_i, v_j) = \begin{cases} -\infty \text{ if } v_i \text{ and } v_j \text{ are in same frame} \\ \alpha lex(v_i, v_j) + (1-\alpha) syn(v_i, v_j) \text{ otherwise.} \end{cases}$$

Only execute merge if confidence is above threshold θ. Initialize θ at 1 and decrement 0.0025 for each iteration. Terminate when θ reaches ⅓ (experimentally determined value).

Evaluation 1

$$PU = \frac{1}{N}\sum_i \max_j |G_j \cap C_i| \qquad CO = \frac{1}{N}\sum_j \max_i |G_j \cap C_i| \qquad F_1 = \frac{2 \times CO \times PU}{CO + PU}$$

	Syntactic Function			Latent Logistic			Split-Merge			Graph Partitioning		
	PU	CO	F1	PU	CO	F1	PU	CO	F1	PU	CO	F1
auto/auto	72.9	73.9	73.4	73.2	**76.0**	74.6	81.9	71.2	**76.2**	**82.5**	68.8	75.0
gold/auto	77.7	**80.1**	**78.9**	75.6	79.4	77.4	**84.0**	74.4	**78.9**	**84.0**	73.5	78.4
auto/gold	77.0	71.0	73.9	77.9	**74.4**	76.2	86.5	69.8	**77.3**	**87.4**	65.9	75.2
gold/gold	81.6	**77.5**	79.5	79.5	76.5	78.0	**88.7**	73.0	**80.1**	88.6	70.7	78.6

Table 1: Evaluation of the output of our graph partitioning algorithm compared to our previous models and a baseline that assigns arguments to clusters based on their syntactic function.

Evaluation 2

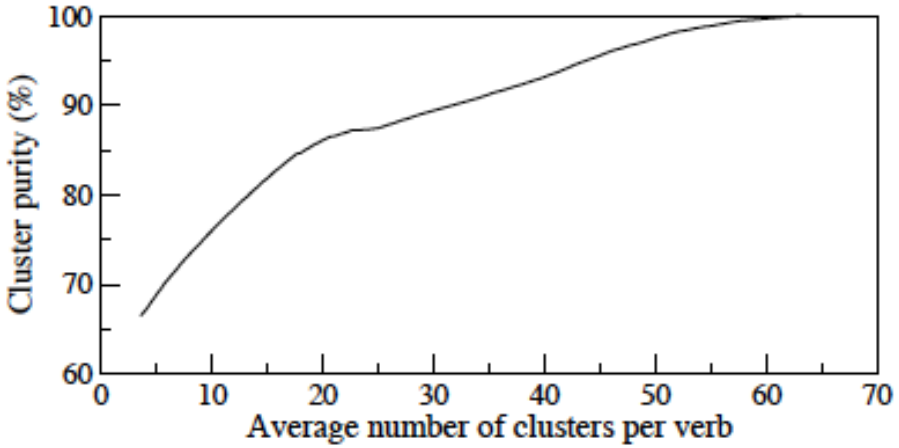


Figure 3: Purity (vertical axis) against average number of clusters per verb (horizontal axis) on the auto/auto dataset.

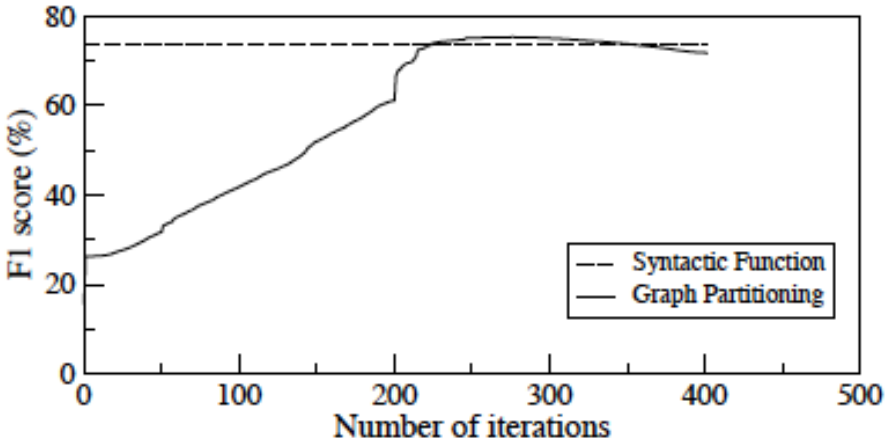


Figure 4: F1 (vertical axis) against number of iterations (horizontal axis) on the auto/auto dataset.

Evaluation 3

Graph Partitioning												
PU	95.6	83.5	72.3	75.4	83.3	84.4	74.8	84.8	89.5	83.0	73.2	66.3
CO	89.1	62.7	42.1	64.2	56.2	66.3	57.2	73.2	64.1	54.3	66.0	57.7
F1	92.2	71.6	53.2	69.4	67.1	74.3	64.8	78.5	74.7	65.7	69.4	61.7
Verb	say	make	go	increase	know	tell	consider	acquire	meet	send	open	break
Freq	15238	4250	2109	1392	983	911	753	704	574	506	482	246

		Syntactic Function			Split-Merge		
Verb	Freq	PU	CO	F1	PU	CO	F1
say	15238	91.4	91.3	**91.4**	93.6	81.7	87.2
make	4250	68.6	71.9	70.2	73.3	72.9	**73.1**
go	2109	45.1	56.0	49.9	52.7	51.9	**52.3**
increase	1392	59.7	68.4	63.7	68.8	71.4	**70.1**
know	983	62.4	72.7	**67.1**	63.7	65.9	64.8
tell	911	61.9	76.8	68.6	77.5	70.8	**74.0**
consider	753	63.5	65.6	64.5	79.2	61.6	**69.3**
acquire	704	75.9	79.7	77.7	80.1	76.6	**78.3**
meet	574	76.7	76.0	76.3	88.0	69.7	**77.8**
send	506	69.6	63.8	66.6	83.6	65.8	**73.6**
open	482	63.1	73.4	67.9	77.6	62.2	**69.1**
break	246	53.7	58.9	56.2	68.7	53.3	**60.0**

Table 3: Clustering results for individual verbs with our split-merge algorithm and the syntactic function baseline.

Evaluation 4

Role	Syntactic Function PU	CO	F1	Split-Merge PU	CO	F1
A0	74.5	87.0	80.3	79.0	88.7	**83.6**
A1	82.3	72.0	76.8	87.1	73.0	**79.4**
A2	65.0	67.3	66.1	82.8	66.2	**73.6**
A3	48.7	76.7	59.6	79.6	76.3	**77.9**
ADV	37.2	77.3	**50.2**	78.8	37.3	**50.6**
CAU	81.8	74.4	**77.9**	84.8	67.2	75.0
DIR	62.7	67.9	**65.2**	71.0	50.7	59.1
EXT	51.4	87.4	64.7	90.4	87.2	**88.8**
LOC	71.5	74.6	**73.0**	82.6	56.7	67.3
MNR	62.6	58.8	**60.6**	81.5	44.1	57.2
TMP	80.5	74.0	**77.1**	80.1	38.7	52.2
MOD	68.2	44.4	53.8	90.4	89.6	**90.0**
NEG	38.2	98.5	55.0	49.6	98.8	**66.1**
DIS	42.5	87.5	57.2	62.2	75.4	**68.2**

Table 4: Clustering results for individual semantic roles with our split-merge algorithm and the syntactic function baseline.

Conclusion

- Both models use a strategy that merges arguments into clusters that hopefully represent semantic roles.
- The results from this unsupervised system show significant promise.
- In order to add to PropBank or some other database, it would still take a lot of human effort.

Discussion questions

1. Are there any heuristics that Lang and Lapata use that you find unfeasible?
2. What is more important for the task of role induction: purity or collocation?
3. Relatedly, what is the ideal granularity for semantic role labels?
4. Is it better to have a set of roles for each verb (PropBank) or to group verbs with similar sets of role together (FrameNet)?