Automatic fine grained semantic classification for domain adaptation

Maria Liakata†, Stephen Pulman††
†Department of Computer Science, University of Wales, Aberystwyth
††Computing Laboratory, University of Oxford
†mal@aber.ac.uk, ††sgp@clg.ox.ac.uk

STEP 2008, Venice

Deriving domain-specific classes for verb arguments

- **Goal:** Obtain domain-specific semantic classes for verbs & their arguments
- **Why?**
  - Verb arguments important in NLP (WSD, ambiguity resolution) e.g. 'Flying planes can be dangerous' vs 'Swallowing apples can be dangerous'
  - WordNet & FrameNet often unable to cater for domain-specific senses
- **Our hypothesis:** Better to induce verb sense & semantic types automatically from the data of domain of interest
- **How:** Cluster verbs & their arguments simultaneously

Outline of the Talk

- Background to semantic classification
- Method for clustering verbs & their arguments to obtain semantic classes
- Interpretation of the semantic classes
- Results & Evaluation
- Future Work

Related Work

- Literature on acquiring semantic classes extensive
- Mainly motivated by WSD, clustering nouns or verbs
- Most relevant to our work:
  - [SchulteimWalde 2003] method for clustering German verbs by linguistically motivated feature selection
  - [Korhonen et al 2006] cluster verbs from biomedical domain
  - [Gamallo et al 2005] perform dual clustering of words and their lexico-syntactic contexts. Create lexicon of words & requirements applied to PP by clustering similar syntactic positions.
  - [Pustejovsky et al 2004] combine selection contexts for verbs to form CPA patterns semi-automatically.

Investigation into automated verb induction

- Do syntactic/semantic analysis of corpus for predicate-argument identification

Investigation into automated verb induction

- Do syntactic/semantic analysis of corpus for predicate-argument identification
- For a given verb, find head nouns occurring as subj, obj, iobj

Investigation into automated verb induction

- Do syntactic/semantic analysis of corpus for predicate-argument identification
- For a given verb, find head nouns occurring as subj, obj, iobj
- Cluster the verb argument slots together according to shared filler nouns $\longrightarrow$

Investigation into automated verb induction

- Do syntactic/semantic analysis of corpus for predicate-argument identification
- For a given verb, find head nouns occurring as subj, obj, iobj
- Cluster the verb argument slots together according to shared filler nouns →
- Noun clusters characterising semantic types of argument slots

Investigation into automated verb induction

- Do syntactic/semantic analysis of corpus for predicate-argument identification
- For a given verb, find head nouns occurring as subj, obj, iobj
- Cluster the verb argument slots together according to shared filler nouns $\longrightarrow$
- Noun clusters characterising semantic types of argument slots
- Side effect: clustering verbs with similar slot

Investigation into automated verb induction

- Do syntactic/semantic analysis of corpus for predicate-argument identification
- For a given verb, find head nouns occurring as subj, obj, iobj
- Cluster the verb argument slots together according to shared filler nouns →
- Noun clusters characterising semantic types of argument slots
- Side effect: clustering verbs with similar slot
- Verb class induction: e.g. *'admit'*, *'deny'* clustered together if their arg share the same filler words (e.g. obj *'wrongdoing'*)

The Corpus & pre-processing

- **Domain of application:** Financial News
- **Corpus:** WSJ section of Penn Treebank II
- **Why?** Predicate-argument structures easily accessible.
- **Corpus Statistics:** 2454 articles (300,000 words), 2798 distinct verb predicates
- **Pre-processing:**
  - Obtained predicate-argument structures using [Liakata & Pulman 2002].
  - Boosting of low frequency verbs
  - Merging together arguments that are NEs: person names, companies, locations, numeric expressions etc.

Clustering argument slots of verbs (1)

We assume: Argument slots of predicates can be characterised by their filler words like a document is characterised by the words it contains.

VERB-ARG	FILLER WORDS/FREQ
invest-subj	person-394,company-86,investor-29,fund-20,..
invest-obj	money-204,person-172,percentage-80,price-36,..
invest-iobj	proposition-63,share-3,money-2,loan-2,..
give-subj	person-7519,company-1889,analyst-296,location-211,..
give-obj	person-605,percentage-350,money-261,agreement-86,..
give-iobj	proposition-610,person-6,money-4,offer-3,..

Therefore: To cluster verb-argument slots together, represent them using Vector Space Model (VSM) & compare their filler words

Clustering argument slots of verbs (2)

VERB-ARG	freq of FILLER WORDS as features			
	person	company	analyst	percentage ..
invest-subj	394	86	13	4..
invest-obj	173	43	0	82..
invest-iobj	1	0	0	0..
give-subj	7519	1889	296	43 ..
give-obj	605	45	9	350..
give-iobj	6	2	0	0

- A matrix containing all verb-arg slots (8,394) as rows and all possible word fillers (32,990) as columns is very sparse.
- Feature selection is required to reduce size of matrix.
- Clustering using **Autoclass**

Autoclass system

- [Cheeseman & Stutz 1995] is probabilistic clustering method.
- Autoclass is an extension of the mixture model as each instance can be characterised by multiple attributes
- Assumes instances of each cluster follow probability distribution
- Clustering problem is given number of clusters find the parameters of the distributions
- Input data in matrix format
- **Why Autoclass?**
  - Number of clusters/classes unknown
  - Probabilistic membership to multiple classes allowed

Clusters & their interpretation

- Best result: 32 classes, all verb-arg assigned deterministically
- **Class measures:** *strength, weight, cross-entropy*
- **Most influential features:** *the ones corresponding to precise concepts associated with specific contexts (freq*idf)*
- *Look at class members to interpret classes:*

```
class(9,[‘climb_arg1’,‘add_up_arg1’,‘shoot_arg1’,‘balloon_arg1’,
        ‘deflate_arg1’,‘decline_arg1’,‘crash_arg1’,
        ‘sink_arg1’,‘slump_arg1’,‘blossom_arg1’,‘blow_arg1’,
        ‘blow_up_arg1’,‘come_up_arg1’,‘boost_arg2’,
        ‘set_off_arg1’,‘break_arg1’,‘soar_arg1’,‘come_down’’,
        ‘slip_arg1’,‘bud_arg1’,‘build_up_arg1’,
        ‘bump_up_arg1’,..)
```

- Class 9 as **group of verbs**: Verbs showing sudden movement and numeric change.
- Class 9 as implicit **group of nouns**:‘Financial indicators’.

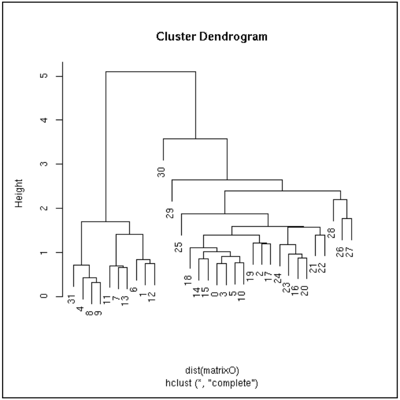
From noun clusters to semantic typing

- Nouns arg to several verbs therefore belong to more than one class. Look at **tf-idf** for most representative class for a term
- **Interpretation for each class:** highest ranking items by descending tf-idf

class	label
0	proposition
1	company_organisation
2	unspecified_someone
3	proposition_truth_profit_patient_impact
4	percentage_money_income_revenue_stock_share_asset
5	percentage_mony_numXpression
6	spokesman_company_person_analyst
7	income_revenue_net_rate_cost_stock
8	place_step_effect_loss_action
9	proposition_company_spokesman_revenue_analyst
10	proposition_stake_rate_percentage
11	proposition_percentage_sure_decision_bid
12	year_percentage_quarter_index
13	reporter_dividend_money_percentage_analyst
14	percentage_proposition_numXpression
15	proposition
16	percentage_stake_demand_money_rate_cash_capital
17	proposition_projection_rate
18	proposition_trading_pressure
19	proposition_table_corner_board_tide
20	proposition_percentage_public_private_high_low
21	government_civilian_unspecified
22	proposition_unspecified_game_role_cash_company
23	percentage_proposition_numXpression
24	percentage_proposition_date_profit
25	director_court_partner_company
26	proposition_contract_profit_demand_requirement
27	demand_problem_leak
28	year_month_time
29	proposition_money_percentage_share_stock
30	year_time
31	fund_proposal_investor

Hierarchy for semantic typing

- Obtaining semantic type/labels for classes non-trial because of overlap
- Hierarchical clustering with overlap coefficient: $sim(A, B) = \frac{|A \cap B|}{min(|A|, |B|)}$



From semantic classes to patterns

- To facilitate the use & evaluation of classes for semantic type assignment, we automatically created verb patterns:

ARG1 **VERBv** (ARG2) (ARG3)

- 'One sense per corpus' assumption, one pattern for each verb
- Patterns modelled on CPA patterns [Pustejovsky et al 2004]
- For example:
  1 report 4 (10) equivalent to:

  [*company_organisation*] **report**
  [*percentage_money_income_revenue_stock_share_asset*]
  [*proposition_stake_rate_percentage*]

Example: patterns used to assign semantic types

Example: *'That is the first time both indexes dropped by double-digit percentages.'*

- **Text to assign semantic type to:** *'indexes dropped by percentages'*
- **relevant pattern:** *[9 drop 8 28]*
- **Check:** *Does 'index' have class 9? Does 'percentage' have class 28?*
- **Check:** *How 'close' are class 9 & the actual class of 'index'?*

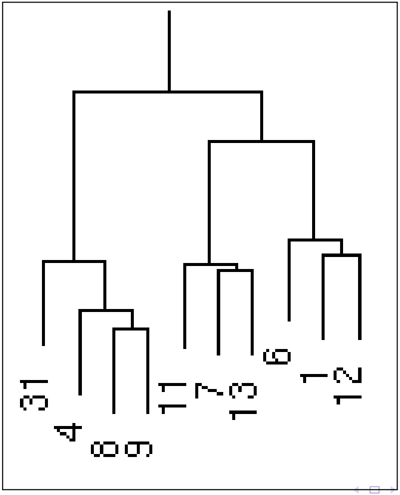
Algorithm for the evaluation of semantic patterns

- Preliminary evaluation on two articles **WSJ, FT** March 2008
- Parsed the articles using CC-Tools, obtained **subj**,**obj** and **iobj** dependencies → evaluation set

For each verb-argument pair token in the evaluation set:

1. Look for a pattern in the database for that verb (Recall cnt + 1)
2. Obtain the type that the pattern assigns to the argument
3. Get the correct type (3 with highest freq out of 10 with highest tf-idf)
4. If type assigned matches any of the 3 classes-semantic types,assignment correct.
5. Otherwise look at cluster dendrogram and find distance btw correct and returned types.
6. Proceeded to the next verb-argument pair.

Subsection of the class dendrogram



Example: evaluation of assignment

- **Example 1:** *'The index is calculated using mortgage loans of $417,000 or less.'*
- **Example 2:** *'Ofheo oversees the government-sponsored mortgage-finance companies Fannie Mae and Freddie Mac'.*

RASP-like dependencies (ncsubj, dobj, iobj) generated by CC-tools:

```
dobj using_4 loans_6
dobj oversees_1 Mae_7
ncsubj oversees_1 Ofheo_0
dobj oversees_1 Mac_10
```

The patterns: Ex1: [6 use **4** 14]   Ex2: [**12** oversee **13** 15]

- Ex1: For 'loan' the correct class is (4,7,11) -**Correct!**
- Ex2: For 'Ofheo' the correct class is (7,12,4) -**Correct!**
- Ex2: For 'Mac','Mae' the correct class is (6,9,1) -*Wrong.*

Results

	verbs	verb-arg	recall	exact match
WSJ	46	78	78/78	33/78 (43%)
FT	24	53	53/53	21/53 (39.6%)

	distance 1	distance 2	distance 3
WSJ	41/78 (53%)	55/78 (70.5%)	60/78 (76.9%)
FT	26/53 (49%)	30/53 (56.6%)	33/53 (62.2%)

Conclusion & Future Work

- Method for for **automatically acquiring domain-specific selectional restrictions for verbs**
- Promising initial results
- Extend to biomedical domain
- Obtain parses and LFs for new texts (using CC-tools and Boxer)
- Try different clustering method and feature selection

References

P. Cheeseman and J. Stutz
Bayesian classification (AutoClass): Theory and results.
Advances in Knowledge Discovery and Data Mining
AAAI Press, 153–180, Menlo Park, CA

P. Gamallo, A. Agustini and G.P. Lopes
Clustering Syntactic Positions with Similar Semantic Requirements
Computational Linguistics, 31(1), 107-146

A. Korhonen, Y. Krymolowski, N. Collier
Automatic Classification of Verbs in Biomedical Texts
Proceedings of AC-COLING 2006, Sydney, Australia

References

M. Liakata and S.G. Pulman
Learning Theories from Text
COLING 2004, Geneva, Switzerland

J. Pustejovsky, P. Hanks and A. Rumshisky
Automated Induction of Sense in Context
COLING 2004, Geneva, Switzerland, 55-58

S.Schulte im Walde
Experiments on the Choice of Features for Learning Verb Classes
Proceedings of EACL 2003, Budapest, Ungarn

Acknowledgements

Thanks to Rachele de Felice for her assistance, Stephen Clark & Rada Mihalcea for their useful comments.

This work was partially funded by:

- the Companions project (`http://www.companions-project.org`)
- the ART Project (`http://www.aber.ac.uk/compsci/Research/bio/art/`)