COSI137B P2: Coreference Resolution

Aaron Levine, Tianzhi Chen, Zachary Yocum

Task

- Given pairs of named entities, identify if they are links in a coreference chain
  - Have access to named entity types
  - Surrounding tokens (questionable tokenization)
  - Part of speech tags
  - Tried to get useful syntax tree data, but provided tokenization prevented us from getting useful features

Dataset

Issue	• 91653 pairs in test • 73049 pairs in dev • 46242 pairs in train (This size order should be reversed, ideally.)	Base tokenization • Odd treatment of punctuation • Null tokens • Sentence segmentation included extraneous data, such as headlines
Result	Hard to train a model that will generalize	We gave up attempting to get syntactic parse tree features
Solutions	• Treat dev+test as train and treat train as test • Cross-validation	• We tried both the Charniak and Stanford parsers • Ultimately we found no solution

Issues with syntactic parsing

- Charniak
  - Uses it's own tokenization solution
  - Examples:
    - system tokenizes ‘ “ ’, but dataset **DIDN'T** Found over 50% of documents contain quotation marks
    - System ignores “empty” tokens that were provided in pos-tagged files like “__VBZ” (is “_” a verb?)
  - Can't go to right place in over 50% trees using index files, so features return false negatives half the time
    - Too much noise to serve as a discriminative feature (scores dropped to ~20% when attmpted)

Issues with syntactic parsing

- Stanford
  - The Stanford parser allowed us to specify the tokenization (after some pre-processing).
  - We used these flags:
    - `-sentences newline`
    - `-tokenized`
    - `-tagSeparator /`
    - `-tokenizerFactory edu.stanford.nlp.process.WhitespaceTokenizer`
    - `-tokenizerMethod newCoreLabelTokenizerFactory edu/stanford/nlp/models/lexparser/englishPCFG.ser.gz`

Issues with syntactic parsing

- Stanford
  - Fails to parse sentences such as:

SEATTLE/NNP _/VBZ The/DT answers/NNS to/TO the/DT three/CD most/RBS commonly/JJ asked/VBD questions/NNS at/IN the/DT Woodland/NNP Park/NNP Zoo/NNP in/IN Seattle/NNP these/DT days/NNS are/VBP :/: one/CD ,/, 235/CD pounds/NNS ;/: two/CD ,/, 22/CD months/NNS (the/IN longest/JJS pregnancy/NN of/IN any/DT mammal/NN in/IN the/DT world/NN);/IN three/CD ,/, natural/JJ insemination/NN ,/, after/IN her/PRP$ 8,800-pound/NN mother/NN was/VBD transported/VBN 2,000/CD miles/NNS for/IN a/DT tryst/NN in/IN Missouri/NNP ./.

Issues with syntactic parsing

- Stanford
    - Parses the sentence if **SEATTLE/NNP _/VBZ** is removed:

```
(ROOT (S (NP (NP (DT The) (NNS answers)) (PP (TO to) (NP (DT the) (CD three)))) (ADVP (RBS most) (RB commonly)) (VP (VBD asked) (SBAR (S (NP (NP (NNS questions)) (PP (IN at) (NP (NP (DT the) (NNP Woodland) (NNP Park) (NNP Zoo)) (PP (IN in) (NP (NNP Seattle)))))) (NP (DT these) (NNS days)) (VP (VBP are) (: :) (NP (NP (CD one)) (, ,) (NP (NP (CD 235) (NNS pounds)) (: ;) (NP (NP (CD two)) (, ,) (NP (NP (CD 22) (NNS months)) (NP (NP (ADJP (JJ (the) (JJS longest)) (NN pregnancy)) (PP (IN of) (NP (NP (DT any) (NN mammal)) (PP (IN in) (NP (DT the) (NN world) (QP (CD );) (CD three)))))))))) (, ,) (NP (JJ natural) (NN insemination)) (, ,) (SBAR (IN after) (S (NP (PRP$ her) (JJ 8,800-pound) (NN mother)) (VP (VBD was) (VP (VBN transported) (NP (CD 2,000) (NNS miles)) (PP (IN for) (NP (NP (DT a) (NN tryst)) (PP (IN in) (NP (NNP Missouri))))))))))))) (. .)))
```

Issues with syntactic parsing

- Stanford
  - Possible to produce parse trees if we exclude certain tokens
  - **However**
    - We’re not confident we could even automate the detection and removal of these extraneous tokens
    - Mention pair data are indexed on the given tokenization
    - Our parse trees would be based on different tokenization/segmentation
    - We’d be ultimately be back to the same problem as with the Charniak parser

Dataset

- Training data was somewhat sparse
  - 46242 Mention pairs
  - 3846 Coreferential pairs
- Devset
  - 73049 Mention pairs
  - 1198 Coreferential pairs
- Testset
  - 91653 Mention pairs
  - 1597 Coreferential pairs

Dataset

- For development training / devset
  - 40 / 60 % split on full data (very bad)
  - 75 / 25 % split on "yes" label (80 / 20 preferable)

- For final testing training + devset / testset
  - 55 / 45 % split on full data (kinda bad)
  - 75 / 25 % split on "yes" label (80 / 20 preferable)

Weka

- Java library of classification models
  - Naive Bayes models
  - Logistic Regression models
  - SVMs
  - etc.
- GUI for feature ranking, removing features, creating different test / training sets, and so on.

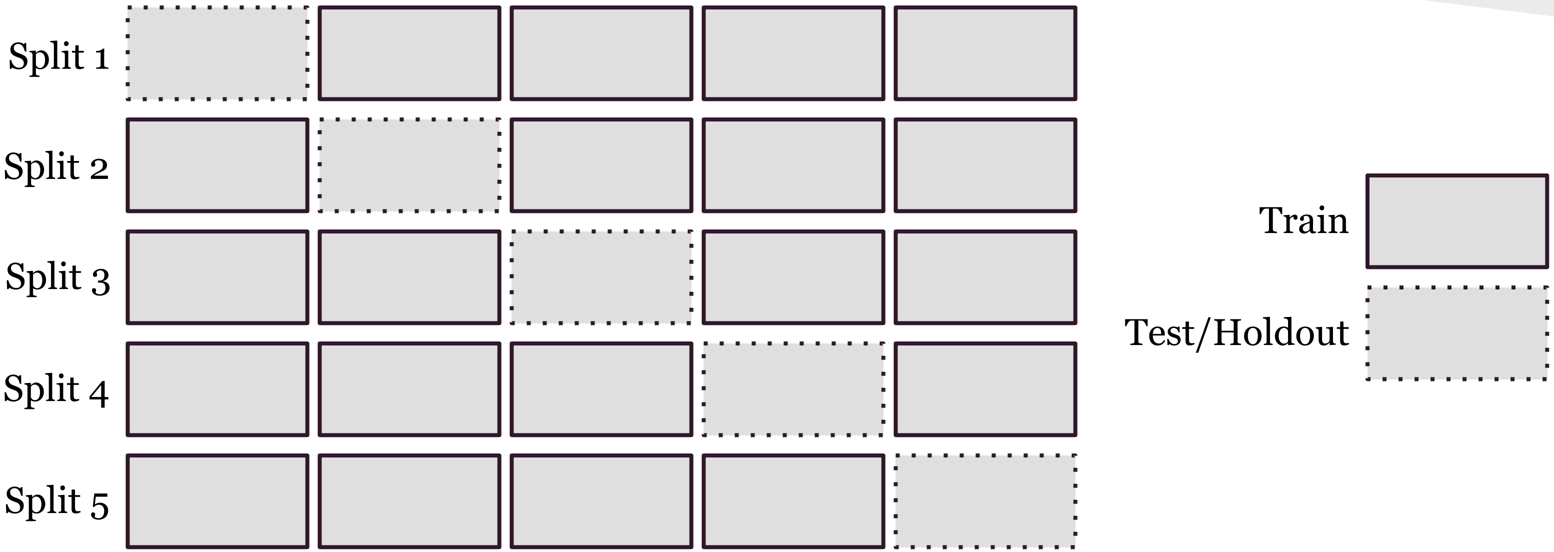
K-fold cross-validation

- Cross-validation is a way to maintain separation between train/test data, but potentially get a better sense of performance on a small data set.
- Takes *k* mutually exhaustive, pair-wise disjoint partitions of the data and treat each as a train:test split.
- Each split has a distinct train:test partition of size

$$\frac{k-1}{k} : \frac{1}{k}$$

- Take the mean performance over all *k* splits.

5-fold cross-validation



19 Features

Name	**Type**	**Description**
`string_match`	`bool`	Are the texts of both mentions identical?
`token_match`	`bool`	Is at least one token an exact match between both mentions?
`entity_type_match`	`bool`	Are the entity types for both mentions identical?
`pos_match`	`int`	How many POS tags match between both mentions?
`number_match`	`bool`	Heuristically, do the mentions match in terms of grammatical number?
`simple_pos_match`	`bool`	Simplified POS-based match based on first character in the POS tags.
`appositives`	`bool`	Does the mention pair fuzzily match the form “a , b”?
`predicative_nominative`	`bool`	Does the mention pair fuzzily match the form “a (is|are|was|were) b”?
`relative_pronoun`	`bool`	Does the mention pair fuzzily match the form “a (which|who|that) b”?
`acronym`	`bool`	Is mention a potential acronym or initialism for mention b or vice versa?

19 Features

Name	Type	Description
string_match_lower	bool	Are the texts of both mentions identical in lowercase?
sentence_distance	int	are mentions in same sentence? or n sentences away?
simple_token_distance	int	how does the positioning of each mention compare to each other
extended_token_distance	int	naive joining of sentence and token distance features.
extend_pos_match	int	How many POS matches occur in contexts around both mentions?
extend_simple_pos_match	int	How many simplified POS matches occur in contexts around both mentions?
same_sentence	bool	Do both mentions occur in the same sentence?
substring_match	bool	Is the shorter string a substring of the longer one?
extend_pos_match	int	How many POS matches occur in contexts around both mentions?

Feature Rankings

Name	Chi Squared Score
token_match	19522.74
string_match_lower	18465.57
string_match	17540.83
substring_match	14165.93
extended_token_distance	11880.51
entity_type_match	6353.302
sentence_distance	4741.657
appositives	3167.409
relative_pronoun	2339.402
same_sentence	2211.053

Name	Chi Squared Score
simple_token_distance	1904.869
pos_match	850.5352
predicate_nominative	597.5463
number_match	402.7395
token_inbetween	246.5525
acronym	245.1432
simple_pos_match	237.9226
extend_simple_pos_match	82.1382
extend_pos_match	40.4088

Models

- Naive Bayes
  - Dev: P=0.309 R=0.397 F=0.348
- Sequential minimal optimization
  - popular SVM algorithm built into Weka
  - Dev: P=0.263 R=0.447 F=0.331
- Logistic Regression
  - MaxEnt but with different optimization system
  - Dev: P=0.404 R=0.411 F=0.407

Results (Logistic Regression)

	Precision	Recall	F1
Development	0.404	0.411	0.407
Test	0.591	0.36	0.447
10-fold	0.5	0.568	0.532

Bibliography

1. Holmes, G., Donkin, A., & Witten, I. H. (1994, December). Weka: A machine learning workbench. In *Intelligent Information Systems, 1994. Proceedings of the 1994 Second Australian and New Zealand Conference on* (pp. 357-361). IEEE.
2. Charniak, E. (2000, April). A maximum-entropy-inspired parser. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference* (pp. 132-139). Association for Computational Linguistics.
3. Socher, R., Bauer, J., Manning, C. D., & Ng, A. Y. (2013). Parsing with compositional vector grammars. In *In Proceedings of the ACL conference*.