Improving Bootstrapped Semantic Lexicon Induction

- McIntosh and Curran published a series of papers on bootstrapped semantic lexicon induction for biomedical texts.
- Their work included bootstrapping modifications and explorations of issues that pose challenges for learning semantic lexicons:
  - weighted, strictly mutually exclusive bootstrapping
  - seed word selection
  - negative ("stop") semantic categories
  - semantic drift

Biomedical NLP

- **Biomedical natural language processing** (**BioNLP**) focuses on text mining of scientific articles in biomedicine and molecular biology. [This is different from NLP for clinical medicine.]
- Biomedical texts are filled with specialized terminology, such as gene and protein names, cell types, and biological processes. For example, from a *PLoS Genetics* article:

  USP14 is endogenously expressed in HEK293 cells and in kidney tissue derived from wt mice.

- Genomic research literature is growing rapidly. Teams of human *biocurators* manually index some documents, but they can't keep up. NLP offers the possibility of automated biocuration!
- Specialized NLP tools have been developed for this domain.

PubMed

- PubMed is a free search engine that provides access to a vast amount of biomedical text, including the MEDLINE database.

  From Wikipedia: *As of 3 October 2013, PubMed has over 23 million records going back to 1966, selectively to the year 1865, and very selectively to 1809; about 500,000 new records are added each year.*

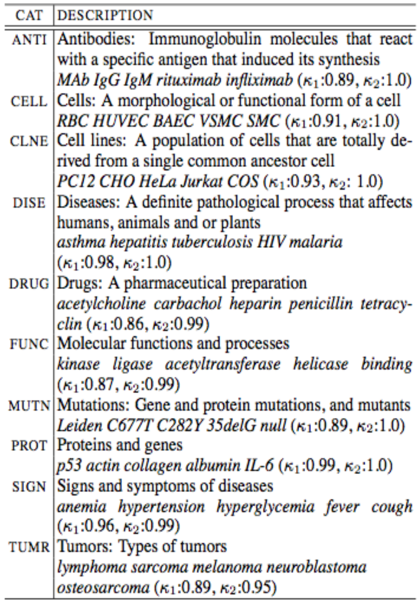
  As of the same date, 13.1 million of PubMed's records are listed with their abstracts, and 14.2 million articles have links to full-text (of which 3.8 million articles are available full-text for free for any user).

- Many scientific articles are indexed with the U.S. National Library of Medicine's Medical Subject Headings (MeSH).

Category definitions and hand-picked seeds

Semantic Categories

Antibodies
Cells
Cell Lines
Diseases
Drugs
Functions/Processes
Mutations
Proteins/Genes
Signs/Symptoms
Tumors



CAT	DESCRIPTION
ANTI	Antibodies: Immunoglobulin molecules that react with a specific antigen that induced its synthesis *MAb IgG IgM rituximab infliximab* (κ_1:0.89, κ_2:1.0)
CELL	Cells: A morphological or functional form of a cell *RBC HUVEC BAEC VSMC SMC* (κ_1:0.91, κ_2:1.0)
CLNE	Cell lines: A population of cells that are totally derived from a single common ancestor cell *PC12 CHO HeLa Jurkat COS* (κ_1:0.93, κ_2: 1.0)
DISE	Diseases: A definite pathological process that affects humans, animals and or plants *asthma hepatitis tuberculosis HIV malaria* (κ_1:0.98, κ_2:1.0)
DRUG	Drugs: A pharmaceutical preparation *acetylcholine carbachol heparin penicillin tetracyclin* (κ_1:0.86, κ_2:0.99)
FUNC	Molecular functions and processes *kinase ligase acetyltransferase helicase binding* (κ_1:0.87, κ_2:0.99)
MUTN	Mutations: Gene and protein mutations, and mutants *Leiden C677T C282Y 35delG null* (κ_1:0.89, κ_2:1.0)
PROT	Proteins and genes *p53 actin collagen albumin IL-6* (κ_1:0.99, κ_2:1.0)
SIGN	Signs and symptoms of diseases *anemia hypertension hyperglycemia fever cough* (κ_1:0.96, κ_2:0.99)
TUMR	Tumors: Types of tumors *lymphoma sarcoma melanoma neuroblastoma osteosarcoma* (κ_1:0.89, κ_2:0.95)

Weighted Mutual Exclusion Bootstrapping (WMEB)

- WMEB is a bootstrapping algorithm that alternately selects patterns and then words for a semantic category.
- WMEB enforces mutual exclusion of semantic categories by discarding words and patterns that are associated with multiple categories.
- Patterns are cumulatively added to a Pattern Pool. The top-k patterns are added in each iteration.
- Words and patterns are ranked based on a **reliability** measure, and ties are broken based on a **relevance weight**.

Word and Pattern Ranking

- Candidate words and patterns are scored based on their **reliability** and **relevance**.
- **Reliability** for a word/pattern is the number of patterns/words that it co-occurs with.
- Relevance is based on the chi-squared (χ^2) measure of statistical significance between a word and pattern.
- The relevance weight is the sum of the χ^2 scores for all pairs:

  for a word, the word is paired with all category patterns

  for a pattern, the pattern is paired with all category words

N-gram Pattern Contexts

- To eliminate the need for syntactic processing, they use **5-grams** (n-gram sequences of size 5).
- Each “pattern” context consists of two words to the left of the target word and two word to its right:

W_{-2} W_{-1} <W> W_1 W_2

For example:

Killing of wild-type <splenocytes> by singly and triply deficient mice …

➡

of wild-type <W> by singly

Data Set Statistics

Type	MEDLINE
Terms	1,347,002
Contexts	4,090,412
5-grams	72,796,760
Unfiltered Tokens	6,642,802,776

Stop (Negative) Categories

- They use **stop categories** to actively learn semantic categories that are not needed for the task.
- Since the learning process assumes mutual, extra categories help to identify ambiguous pattern contexts and draw away words that could be false hits.
- Four stop categories:
  AMINO ACID, ANIMAL, BODY, ORGANISM

Seed Word Sensitivity

- To investigate the impact of the initial seeds on Basilisk and WMEB, experiments used randomly selected "gold" seeds.
- Correct terms were selected either from correct terms extracted by just one algorithm (UNIQUE), or correct terms extracted by both algorithms (UNION).
- Each algorithm was run 10 times, with different randomly selected seeds. The overlap between the 10 generated lexicons was:
  - Top 100 terms: 44% for WMEB, 18% for Basilisk
  - Top 500 terms: 47% for WMEB, 39% for Basilisk
- Basilisk tends to generate esoteric, rare, and misspelled words in the early iterations.

Plotting Seed Word Results

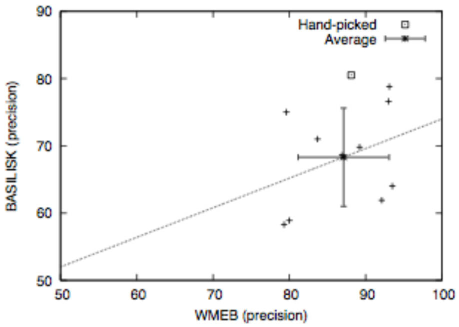
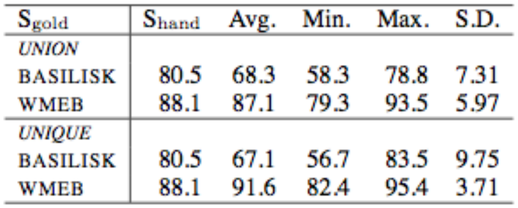


Figure 1: Performance relationship between WMEB and BASILISK on S_{gold} UNION

Performance of Different Seeds



S_{gold}	S_{hand}	Avg.	Min.	Max.	S.D.
UNION					
BASILISK	80.5	68.3	58.3	78.8	7.31
WMEB	88.1	87.1	79.3	93.5	5.97
UNIQUE					
BASILISK	80.5	67.1	56.7	83.5	9.75
WMEB	88.1	91.6	82.4	95.4	3.71

Table 3: Variation in precision with random gold seed sets

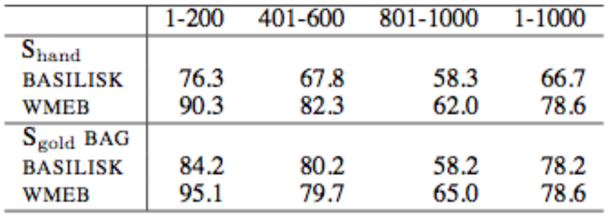
Bagging

- Bagging techniques are ensemble-based methods in machine learning that aggregate results from multiple classifiers.
- Given training data, a set of M training sets are created by uniformly sampling with replacement from the training data.
- M classifiers are then trained, and the predictions of these classifiers are combined (e.g., by voting for classification tasks).
- Bagging tends to reduce variance and alleviate overfitting.

Supervised Bagging

- 50 sets of seeds are generated, by randomly sampling from the UNION evaluation data.
- The bootstrapping algorithm is run 50 times, once with each seed set. This produces 50 lexicons: L_1 … L_{50}
- All terms are then ranked based on the number of lexicons that they appear in. Ties are broken by preferring the term that was learned in the earliest iteration.
- Supervised bagging improved the performance of both algorithms. But it assumes a large set of gold seeds.

Supervised Bagging Results



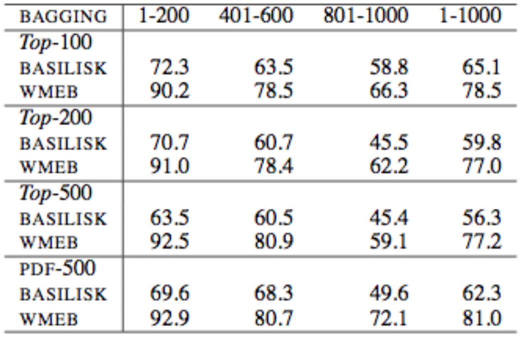
	1-200	401-600	801-1000	1-1000
S_{hand}				
BASILISK	76.3	67.8	58.3	66.7
WMEB	90.3	82.3	62.0	78.6
S_{gold} BAG				
BASILISK	84.2	80.2	58.2	78.2
WMEB	95.1	79.7	65.0	78.6

Table 4: Bagging with 50 gold seed sets

Unsupervised Bagging

- An unsupervised approach creates 50 seed sets by sampling from the lexicon generated from the hand-selected seeds.
- This process involves two rounds of bootstrapping:
  - Induce an initial lexicon from hand-selected seeds
  - Induce a lexicon with unsupervised bagging using seed sets generated from the initial lexicon
- Since the earlier iterations of bootstrapping are usually the most precise, they tried sampling from the top 100, 200, or 500 terms. They also tried the top 500 terms with a bias based on rank.

Unsupervised Bagging Results



BAGGING	1-200	401-600	801-1000	1-1000
Top-100				
BASILISK	72.3	63.5	58.8	65.1
WMEB	90.2	78.5	66.3	78.5
Top-200				
BASILISK	70.7	60.7	45.5	59.8
WMEB	91.0	78.4	62.2	77.0
Top-500				
BASILISK	63.5	60.5	45.4	56.3
WMEB	92.5	80.9	59.1	77.2
PDF-500				
BASILISK	69.6	68.3	49.6	62.3
WMEB	92.9	80.7	72.1	81.0

Table 5: Bagging with 50 unsupervised seed sets

Distributional Similarity

- A common method to assess the *semantic similarity* of words is to compare the contexts in which they occur.

> *"You shall know a word by the company it keeps!"* (Firth 1957)

- **Distributional Similarity** methods compare the contexts that occur around words in a large text collection to determine how similar two words are.

Distributional Hypothesis (Harris, 1954): *words that occur in the same contexts tend to have similar meanings*

Intuition

- I have a gok.
- Julie bought a ***gok***.
- Mark ordered ***gok*** for lunch.
- The ***gok*** seeds fell all over the floor.
- Harry is allergic to ***gok***.
- The ***gok*** wasn't quite ripe yet..
- They planted a rose bush and a ***gok*** tree in their yard.
- The recipe called for beef, ***goks***, and curry paste.

Computing Distributional Similarity

1. Gather all of the contexts around each term.
2. Create a feature vector from the contextual evidence for the term.
3. Compute the similarity of pairs of terms by computing the similarity of their feature vectors.
4. Rank or cluster the most similar terms.

Context

- Context is the neighborhood around an instance of ***w***.
- The neighborhood around ***w*** is typically defined as a ***context window*** of words, phrases, or structures on its left (-) and/or on its right (+).
  - Some tasks use “local” small context windows (e.g., +/- 2 words).
  - Some tasks use “global” large context windows (e.g., +/- 100 words).

Example

CORPUS

She ate chili for lunch.
She went to the park.
She had lunch at a ***diner***.
That ***diner*** serves chili for lunch.
She went shopping at the store.
She had chili at the ***diner***.
For lunch, she went to the ***diner***.

Each row in the table is a *context vector*. (*Context* = sentence in this example.)

Features

	she	**ate**	**chili**	**lunch**	**went**	**park**	**had**	**diner**	**serves**	**shopping**	**store**
she											
…											
diner (bin)	1	0	1	1	1	0	1	0	1	0	0
diner (freq)	3	0	2	3	1	0	2	0	1	0	0
diner (prob)	3/4 .75	0/4 .00	2/4 .50	3/4 .75	1/4 .25	0/4 .00	2/4 .50	0/4 .00	1/4 .25	0/4 .00	0/4 .00
…											
store											

Semantic Similarity

The ***semantic similarity*** of two words is the similarity of their context vectors. Two common similarity metrics are Jaccard and cosine.

The Jaccard similarity metric assesses the amount of overlap between features:

$$\text{Jaccard}(\vec{X},\vec{Y}) = \frac{\sum_{i=1}^{N} \min(x_i , y_i)}{\sum_{i=1}^{N} \max(x_i , y_i)}$$

Weighted Jaccard similarity sums weights instead of just counting.

Semantic Drift

- **Semantic Drift** occurs when the learned words begin to drift away from the original semantic class and represent edge cases or completely different semantic classes.
- Ambiguous words are one cause.
  - *Iris* and *Rose* are both female names and flowers.
  - *April* and *June* are both female names and months.
- Ambiguous contexts are another cause.
  - *He visited his aunt in … Boston … November*
  - *She saw the man on … the beach … Tuesday*

Detecting Semantic Drift

- **Key Idea:** semantic drift has occurred when a candidate word is more similar to recently learned words than to the seeds and (presumably) high precision words learned in early bootstrapping iterations.
- Suppose the current lexicon has size N

$L_{1\ldots n}$ refers to the first n terms added to the lexicon

$L_{(N-m)\ldots N}$ refers to the last m terms added to the lexicon

$$\text{drift}(t, n, m) = \frac{\text{AvgSim}(L_{1\ldots n}, t)}{\text{AvgSim}(L_{(N-m)\ldots N}, t)}$$

Using the drift measure

- Semantic drift detection can be used for post-processing as a filter or incorporated into the learning process.
- In each iteration, the candidate words are ranked & then:
  - If a candidate word has a drift score below a threshold, it is discarded.
  - If a candidate word has zero similarity with the last m terms but is similar to at least one of the first n terms, it is selected.
- The distributional similarity measure uses t-test scores for feature weights and a weighted Jaccard measure as the similarity metric.

Semantic Drift Graph for CELL

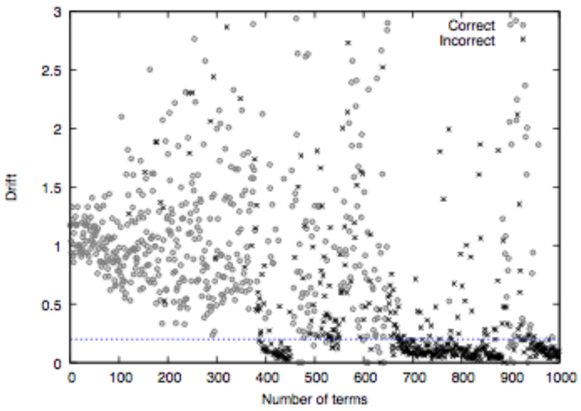
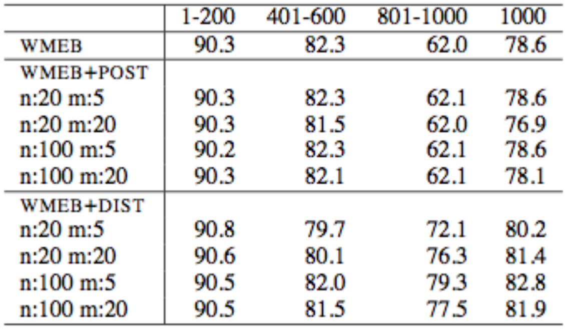


Figure 2: Semantic drift in CELL (n=20, m=20)

Semantic Drift Results

WMEB results using semantic drift detecion as a post-processing filter (POST) or integrated during bootstrapping (DIST).



	1-200	401-600	801-1000	1000
WMEB	90.3	82.3	62.0	78.6
WMEB+POST				
n:20 m:5	90.3	82.3	62.1	78.6
n:20 m:20	90.3	81.5	62.0	76.9
n:100 m:5	90.2	82.3	62.1	78.6
n:100 m:20	90.3	82.1	62.1	78.1
WMEB+DIST				
n:20 m:5	90.8	79.7	72.1	80.2
n:20 m:20	90.6	80.1	76.3	81.4
n:100 m:5	90.5	82.0	79.3	82.8
n:100 m:20	90.5	81.5	77.5	81.9

Table 6: Semantic drift detection results

Summary

- Strictly enforcing mutual exclusion of semantic categories seems to be helpful.

- Bagging with randomly sampled seed sets can help to minimize concerns about suboptimal hand-picked seeds.

- Automatically detecting semantic drift can be effective in improving the quality of a lexicon, especially in the later stages of bootstrapping.

- Having appropriate negative categories (as distractors) can help to draw away potentially confusing words and contexts.