Learning syntactic patterns for automatic hypernym discovery
Rion Snow  Daniel Jurafsky  Andrew Y. Ng
Stanford University

Abstract
We present a new algorithm for learning hypernyms (is-a relations from text, as a key problem in machine learning for natural language understanding. This method generalizes earlier work that relied on hand-built lexico-syntactic patterns by introducing a general-purpose optimization of the pattern space based on syntactic dependency paths. We learn these paths automatically by taking hypernym/hypenym word pairs from WordNet, finding sentence templates containing these words in a large parsed corpus, and automatically extracting these paths. These paths are then used as features in a high-dimensional representation of noun space using a logistic regression classifier based on these features for the task of corpus-based hypernym pair identification. Our classifier is shown to outperform previous pattern-based methods for identifying hypernym pairs (using WordNet as a gold standard), and is shown to outperform those methods as well as WordNet on an independent test set.

Motivation
• It has long been a goal of AI to automatically acquire structured knowledge directly from text, e.g. in the form of a semantic network.

• To date, large-scale semantic networks have mostly been constructed by hand. (e.g. WordNet).

• We present an automatic method for semantic classification that may be used for semantic network construction; this method outperforms WordNet on an independent evaluation task.

Purpose
We aim to classify whether a noun pair (X, Y) participates in one of the following semantic relationships:

- Hypernym: X is a kind of Y.
- Entity-organism > person
- Coordinate (taxonomic sisters)

For every noun pair in a large newswire corpus we use as features 69,592 of the most frequent directed paths (with redundant “silhouettes” of length 1) occurring between noun pairs in MINIPAR syntactic dependency graphs. MINIPAR is a principle-based parser (Lin, 1998), which produces a dependency graph of the form below:

A subsolution of the entity branch in Caraballo’s hierarchy (at right).

Dependency Paths as Features
For every noun pair in a large newswire corpus we use as features 69,592 of the most frequent directed paths (with redundant “silhouettes” of length 1) occurring between noun pairs in MINIPAR syntactic dependency graphs. We learn these paths automatically by taking hypernym/hypenym word pairs from WordNet, finding sentence templates containing these words in a large parsed corpus, and automatically extracting these paths. These paths are then used as features in a high-dimensional representation of noun space using a logistic regression classifier based on these features for the task of corpus-based hypernym pair identification. Our classifier is shown to outperform previous pattern-based methods for identifying hypernym pairs (using WordNet as a gold standard), and is shown to outperform those methods as well as WordNet on an independent test set.

Noun Pairs as Feature Vectors
• Each noun pair is represented as a 69,592-d vector.
• Each entry is the # of times feature f occurs with X.
• Noun pairs labeled as “hypernym” or “not hypernym”.
• WordNet labels provide a training / development set.
• All ancestors allowed as hypernyms – not just direct parents.

Test Sets (Human Labels):

- 50% cross validation on the WordNet-labeled data.
- 10-fold cross validation on the WordNet-labeled data.
- Conclusions: 70,000 features are more powerful than 6.

Building a Semantic Taxonomy

Using this classifier we may now extend and construct semantic taxonomies. We assume that the semantic taxonomy is a directed acyclic graph G. We then consider the set of probabilities \( P(x_1, x_2) \) given by our classifier as noisy observations of the corresponding ancestor relations.

We condition the probability of our observations given a particular DMG \( G \) as:

\[
P(G) = \prod_{i = 1}^{n} P(x_i | x_{parents(i)})
\]

where \( x_i \) is the output for the words \( x_i \) in \( G \).

Our goal is to return the graph that maximizes this probability.

Algorithm: at each step we add the single link \( x_i \rightarrow x_j \) that maximizes the change in probability \( \frac{\partial P(G)}{\partial x_i \rightarrow x_j} \), where:

\[
\frac{\partial P(G)}{\partial x_i \rightarrow x_j} = \sum_{x_k \in \text{children}(x_i)} \left( \frac{1}{P(x_k | x_i)} - \frac{1}{P(x_k | x_j)} \right)
\]

We continue adding links \( x_i \rightarrow x_j \) as long as \( \frac{\partial P(G)}{\partial x_i \rightarrow x_j} > 0 \).

We have begun constructing these extended taxonomies; we plan to release the first of these for use in NLP applications in early 2005. Please let us know if you’re interested in an early release!