Text Mining V: Mining Multifaceted Taxonomies

JIAWEI HAN
COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

JANUARY 30, 2019
Outline

- Taxonomy Basics
- Clustering-based Taxonomy Construction
- TaxoGen: Topic Taxonomy Construction by Adaptive Term Embedding and Clustering
- Instance-based Taxonomy Construction
- HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion
- Summary
What is a Taxonomy?

- Taxonomy is a hierarchical organization of concepts.
- For example: Wikipedia category, ACM CCS Classification System, Medical Subject Heading (MeSH), Amazon Product Category, Yelp Category List, WordNet, and etc.

Wikipedia Category  MeSH  Amazon Product Category  WordNet
Why do we Need a Taxonomy?

- Taxonomy can benefit many knowledge-rich applications
  - Text Understanding
  - Knowledge Organization
  - Document Categorization
  - Recommender System
How to Get a Taxonomy?

- Manual Curation
  - Time-consuming
  - Tremendous human (experts) efforts
- Medical Subject Heading (MeSH): 60+ years
- ACM CCS Classification System: 40+ years
- IEEE Taxonomy: 40+ years

- Automated taxonomy construction from raw text is in great demand

Text Corpus -> User

provide minimal guidance for help
Two Types of Taxonomy

- Clustering-based Taxonomy
- Instance-based Taxonomy
What is a Faceted Taxonomy?

- An example of **faceted taxonomy** in hypersonic vehicle research domain

![Diagram showing a faceted taxonomy with nodes such as root, airbreathing propulsion, thermal management, thermal control system, insulation, active cooling, heat transfer, organic composites, and vascular composites. The diagram includes labels for faceted taxonomy and relations such as “isA” and “used” with descriptions for each facet: Facet 1: [challenges area], Facet 2: [mechanisms of action], Facet 3: [material types].]
Why Faceted Taxonomy?

- Organize, index, and retrieve documents
- Conduct analysis at meaningful levels of abstraction
- Help knowledge extraction and decision making
- Facilitate multi-faceted search
Construction of Faceted Taxonomy

- Manually construct **the whole taxonomy**:
  - Time-consuming
  - Tremendous **human (expert) efforts**
  - WordNet: Over 30 years, starting in 1985
  - ACM Taxonomy: Over 50 years, starting in 1964
  - MeSH Terms: Over 60 years, starting in 1954

- User-guided
- Data-driven
- Knowledge-enhanced

**A better strategy:**
- User provides seeds
- We automate the remaining
Two Kinds of Taxonomy: Instance- vs. Cluster- Based

Instance-based Taxonomy (Is-A)

<table>
<thead>
<tr>
<th>Taxonomy Form</th>
<th>Node</th>
<th>Edge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster-based Taxonomy</td>
<td>A cluster of entities w/o facet info</td>
<td>Statistical correlation</td>
</tr>
<tr>
<td>Instance-based Taxonomy</td>
<td>One entity w/o facet info</td>
<td>Only “isA” relation</td>
</tr>
</tbody>
</table>

(Concept) Cluster-based Taxonomy
What Should Be Done for Taxonomy Discovery: General Thinking

- **Faceted**: Add facet information into taxonomy
  - Each node in taxonomy will be user-interested facets such as “research area”, “application”, and “technique”

- **User-Guided**: Allow user-guidance to assist taxonomy construction
  - User can easily specify the facets and relations among facets by providing seed instances and example relations

- **Application-Oriented**: Exploration of the usage of faceted taxonomy in multiple applications
  - Facilitate faceted document search, cube construction, OLAP, etc.
User-Guidance Facet Completion

- It is easy for a user to provide a few examples to outline her need
  - Ex. 1. Given {Illinois, Maryland}, provide all U.S. states
  - Ex. 2. Given {machine learning, computer architecture}, provide all the disciplines of computer science

- Two critical tasks
  - Set Expansion: Finishing the elements that users in mind but not yet provided
    - Ex. 1. {Illinois, Maryland} → {California, New York, ...} (all 51 states)
    - Ex. 2. {machine learning, computer architecture} → {database, computer systems, algorithms, ...} (all CS disciplines—but hard to be coherent and univ. agreeable)
  - Concept clustering: Clustering concepts around the expanded set elements
    - Ex. 2. {machine learning, SVM, classification, ensemble methods, deep learning, probabilistic graphic models, learning algorithms, ...}
Outline

- Taxonomy Basics
- Clustering-based Taxonomy Construction
- TaxoGen: Topic Taxonomy Construction by Adaptive Term Embedding and Clustering
- Instance-based Taxonomy Construction
- HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion
- Summary
Related Work: Taxonomy Construction: Methodologies

- **Pattern-based Methods**
  - Local Extraction (Hearst Patterns)
    - Find “negative” Hearst patterns
    - Find more “isA” patterns
  - Global Structure Tuning (chaining isA relations)
    - Direct chaining + Heuristically cycle deletion
    - Find Minimum Spanning Tree
    - Find Tree/DAG minimizing the predefined likelihood
  - Using Bayesian Rose Tree
    - Hierarchical agglomerative clustering using LAKI representation

- **(Hierarchical) Clustering-based Methods**
  - Bottom-Up
    - Cathy
  - Top-Down
    - Hierarchical topic modeling
    - nCRP model
    - nHDP model
    - PAM model
Hierarchical Topic Model

- Use a cluster of terms (i.e., a topic) to represent a concept and organize topics in a hierarchical way
- Pose different statistical assumptions on the data generation process
  - Nested Chinese Restaurant Process:
    - hLDA [Blei et al.’03], hLDA-nCRP [Blei et al.’10]
  - Pachinko Allocation Model:
    - PAM [Li and McCallum’06], hPAM [Mimno et al.’07]
  - Dirichlet Forest Model:
    - DF [Andrzejewski et al.’09], Guided HTM [Shin and Moon’17]
Example: hLDA

Assume documents are generated by a nested Chinese Restaurant Process

1. Let $c_1$ be the root restaurant.
2. For each level $\ell \in \{2, \ldots, L\}$:
   (a) Draw a table from restaurant $c_{\ell-1}$ using Eq. (1). Set $c_{\ell}$ to be the restaurant referred to by that table.
3. Draw an $L$-dimensional topic proportion vector $\theta$ from $\text{Dir}(\alpha)$.
4. For each word $n \in \{1, \ldots, N\}$:
   (a) Draw $z \in \{1, \ldots, L\}$ from $\text{Mult}(\theta)$.
   (b) Draw $w_n$ from the topic associated with restaurant $c_z$.

We develop an approach to risk minimization and stochastic optimization that provides a convex surrogate for variance, allowing near-optimal and computationally efficient trading between approximation and estimation error.

"Observed" documents
Assume documents are generated by a mixture of

1. For each document $d$, sample a distribution $\theta_d$ over super-topics and a distribution $\theta_T$ over sub-topics for each super-topic.
2. For each word $w$,
   (a) Sample a super-topic $z_T$ from $\theta_0$.
   (b) Sample a sub-topic $z_t$ from $\theta_{z_T}$.
   (c) Sample a level $l$ from $\zeta_{z_T z_t}$.
   (d) Sample a word from $\phi_0$ if $l = 1$, $\phi_{z_T}$ if $l = 2$, or $\phi_{z_t}$ if $l = 3$.

We develop an approach to risk minimization and stochastic optimization that provides a convex surrogate for variance, allowing near-optimal and computationally efficient trading between approximation and estimation error.

Figure credits to [Mimno et al.’07]
Hierarchical Clustering

- Group terms into hierarchical clusters and each cluster represents an interested concept
- Top-down approaches:
  - CATHY [Wang et al.’13a]
  - CATHYHIN [Wang et al.’13b]
- Bottom-up approaches:
  - BRT [Liu et al.’12] [Song et al.’15]
Example: CATHY [Wang et al.’13a]

- Step 1: Construct the term co-occurrence network using the entire corpus
- Step 2: For each topic, cluster co-occurrence network into subtopic’s sub-networks and estimate each sub-topical phrase’s frequency
  - Use a generative model for clustering and do inference by EM algorithm
- Step 3: For each topic, extract candidate phrases using topical frequency
- Step 4: For each topic, rank topical phrases based on topical frequency
- Step 5: Recursively apply steps 2-5 to each subtopic and construct the hierarchy in a top-down fashion
Example: BRT [Liu et al. ’12]

- Agglomerative multi-branch clustering using Bayesian Rose Tree

Algorithm 1 Bayesian Rose Tree (BRT).

Input: A set of documents $\mathcal{D}$.

$T_i \leftarrow x_i$ for $i = 1, 2, \ldots, n$

$c \leftarrow n$

while $c > 1$ do

1. Select $T_i$ and $T_j$ and merge them into $T_m$ which maximizes $L(T_m) = \frac{p(\mathcal{D}_m|T_m)}{p(\mathcal{D}_i|T_i)p(\mathcal{D}_j|T_j)}$, where the merge operation is join, absorb, or collapse.

2. Replace $T_i$ and $T_j$ with $T_m$ in the tree.

3. $c \leftarrow c - 1$

end while

Join: $T_m = \{T_i, T_j\}$

Absorb: $T_m = \{\text{children}(T_i) \cup T_j\}$

Collapse: $T_m = \{\text{children}(T_i) \cup \text{children}(T_j)\}$
Limitations of Previous Methods

- Too strong statistical assumptions on document generation process
- Bag-of-word document representation ignores word order information
- Real-world data may not follow these statistical distributions/processes
- Computationally slow
- Slow inference restricts their applications to large-scale data
Outline

- Taxonomy Basics
- Clustering-based Taxonomy Construction
- TaxoGen: Topic Taxonomy Construction by Adaptive Term Embedding and Clustering
- Instance-based Taxonomy Construction
- HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion
- Summary
Zhang, Chao, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian M. Sadler, Michelle Vanni and Jiawei Han. “TaxoGen: Unsupervised Topic Taxonomy Construction by Adaptive Term Embedding and Clustering.” KDD 2018

Idea: use term embeddings to construct a clustering-based taxonomy using hierarchical clustering

- Learn term embeddings to capture their semantic correlations
- Construct topic taxonomy in a recursive, top-down fashion
Revisiting Word2Vec: Why Term Embeddings help?

- Word embedding learning\(^{[1]}\) pushes together terms that share same or similar contexts
- Semantically coherent terms are more likely to have close embeddings

\[1\] Mikolov et al., Distributed Representations of Words and Phrases and their Compositionality, NIPS 2013
Visualizations of Term Embeddings on Real Data

Computer Graphics

Cryptography
Taxonomy Generation from Massive Text Corpora

- Automated construction of topic taxonomy
- Selected method: **spherical clustering**—Use **embeddings** to find semantically consistent clusters
  - Domain-specific terms can be clustered together
    - “machine learning”, “learning algorithm”, ...
  - Where do the general terms go?
    - “computer science”, “method”, “paper”, ...

![Diagram showing the construction of a topic taxonomy from documents](image)

**Documents**

**Topic Dimension**

- computer_science
- data_minning
- networking
- computer_vision
- machine_learning
- information_retrieval

- clustering
- reinforcement_learning
- deep_learning
- bayesian_network
- classification

- **CS**
  - CG
  - ML
  - IR

- **Clustering**
- **Classification**

*recursive construction*
Challenge 1: Determining the Proper Levels for Terms

- Not every term should be pushed down to child levels
  - When splitting a parent topic, general terms (e.g., computer science) should remain in the parent level
- How to determine which terms should be kept in the parent node?
Solution: Adaptive Term Clustering with Spherical K-Means

- Design a ranking module to select **representative phrases** for each cluster
- Conduct comparative analysis (combining **popularity** and **concentration**)
  - Does this phrase better fit my cluster or my siblings‘?
- Push the **background phrases** back to the general node
  - “computer science”, “paper” → the higher-level node (root node)
  - “machine learning”, “ml”, “classification“ → the “ML” node
- The set of remaining phrases leads to more separable clustering

After pushing up general terms, the remaining terms become more separable
Solution: Adaptive Term Clustering with Spherical K-Means

Algorithm 1: Adaptive clustering for topic splitting.

Input: A parent topic $C$; the number of sub-topics $K$; the term representativeness threshold $\delta$.

Output: $K$ sub-topics of $C$.

1. $C_{\text{sub}} \leftarrow C$;
2. while True do
3.  $S_1, S_2, \ldots, S_K \leftarrow \text{SPHERICAL-K-MEANS}(C_{\text{sub}}, K)$;
4.  for $k$ from 1 to $K$ do
5.     for $t \in S_k$ do
6.         $r(t, S_k) \leftarrow$ representativeness of term $t$ for $S_k$;
7.         if $r(t, S_k) < \delta$ then
8.             $S_k \leftarrow S_k - \{t\}$;
9.     $C'_\text{sub} \leftarrow S_1 \cup S_2 \cup \ldots \cup S_K$;
10.    if $C'_\text{sub} = C_{\text{sub}}$ then
11.        Break;
12.    $C_{\text{sub}} \leftarrow C'_\text{sub}$;
13. Return $S_1, S_2, \ldots, S_K$;

- Measuring term representativeness:

$$ r(t, S_k) = \sqrt{\text{pop}(t, S_k) \cdot \text{con}(t, S_k)} $$

**Popularity:**

$$ \text{pop}(t, S_k) = \frac{\log(tf(t, D_k) + 1)}{\log tf(D_k)} $$

**Concentration:**

$$ \text{con}(t, S_k) = \frac{\exp(\text{rel}(t, D_k))}{1 + \sum_{1 \leq j \leq K} \exp(\text{rel}(t, D_j))} $$
Challenge 2: Learning Discriminative Embeddings

- Term embeddings have dampening discriminativeness at lower levels
- The discriminative power of embeddings becomes weaker
- Unclear cluster structures due to “insufficient training”
Solution: Local Embedding for Discriminative Power

- Global embedding (embedding learning on the global dataset) does not work
  - Terms at different granularity may have close embeddings
  - Ex. “Information Extraction”: similar to “text mining”, “NLP”, “machine learning”
- Solution: Local embedding:
  - For each “sub-topic” node, learn local embedding only on relevant documents
  - Only preserve information relevant to the “sub-topic”
- How to collect relevant documents for each child topic?
  - Use term clustering results to infer the cluster memberships for documents

With local embedding
  - 1. Clearer parent-child relations
  - 2. More separable clusters
Experimental Settings

- **Datasets**
  - DBLP: paper titles for 1.8 M DBLP papers, 13 K extracted terms
  - SP: Paper abstracts for 94 K signal processing papers, 7 K extracted terms

- **Task**
  - DBLP: construct a 4-level topic taxonomy with the terms, # of branches = 5
  - SP: construct a 3-level topic taxonomy with the terms, # of branches = 5

- **Evaluation metrics**
  - Relation accuracy: measure the quality of parent-child relations
  - Term coherency: measure the semantic coherency of terms in a topic
  - Cluster quality: measure the separability of sibling clusters
TaxonGen: Adaptive Spherical Clustering + Local Embedding

- Phrase mining + Adaptive spherical clustering: Generate top-level clusters
- Local embedding: Generate lower level clusters

Experiment with the DBLP dataset

High quality multi-level hierarchy generated automatically
Experiment with the DBLP dataset

Part of the generated taxonomy
Part of the subtree on Machine Learning

- Neural Net
  - neural_network
    - neural_network
      - model_predictive_control
      - neuron
      - controller
    - neurons
      - adaptive_control
      - nonlinear_systems
      - sliding_mode
  - kernel_discriminant_analysis
    - kernels
    - semidefinite_programming
    - regularization
    - sparsity
    - low_rank
    - nonnegative_matrix_factorization
    - regularized

- Kernel Methods
  - RL
    - reinforcement_learning
      - optimal_control
      - stochastic
      - markov_decision_processes
      - discrete_time
      - integer_programming
      - optimisation
      - continuous_time

- Classification
  - classifiers
    - classifiers
    - rough_sets
    - rough_set
    - biclustering
    - cancer
    - rough_set_theory
    - decision_tree
    - attribute_reduction

- Bayesian Model
  - bayesian
    - bayesian
    - tree_automata
    - causal
    - belief_propagation
    - mcmc
    - bounds
    - em
    - maximum_likelihood

- Fuzzy Control
  - tsk_fuzzy
    - genetic_algorithms
    - controller
    - fuzzy_control
    - fuzzy_controller
    - genetic_algorithm
    - model_predictive_control
    - nonlinear_systems

- RNN
  - recurrent_networks
    - neuronal
    - dynamical
    - synaptic
    - perturbation
    - neural
    - chaotic
    - synapses

- SOM
  - som
    - self_organizing_maps
    - selforganizing_map
    - fuzzy_clustering
    - ant_colony
    - kohonen
    - incremental_learning
    - soms

- Forecast
  - forecasting
    - independent_component_analysis
    - speaker_recognition
    - ann
    - stock_market
    - non_linear

- Back Propagation
  - back_propagation_bp
    - back_propagation_bp
      - cascade
      - rbf
    - radial_basis_functions
      - mlp
      - radial_basis_function_networks
      - rbf_network
      - rbf_networks
Global Embedding vs. Local Embedding

- Use embeddings to perform top-K queries
- Local embeddings can capture semantic similarity more accurately at lower levels

<table>
<thead>
<tr>
<th>Query</th>
<th>Global Embedding</th>
<th>Local Embedding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1: pose_estimation</td>
<td>pose_estimation, single_camera, monocular, d_reconstruction, visual_servoing</td>
<td>pose_estimation, camera_pose_estimation, dof, dof_pose_estimation, uncalibrated</td>
</tr>
<tr>
<td>Q2: information_extraction</td>
<td>information_extraction, information_extraction_ie, textMining, named_entity_recognition, natural_language_processing</td>
<td>information_extraction, information_extraction_ie, extracting_information_from_question_answering_qa</td>
</tr>
</tbody>
</table>
Quantitative Performance Based on Human Evaluation

- **Relation accuracy**: the accuracy of parent-child relationships
  - Judge whether each edge in the topic taxonomy has valid parent-child relation
- **Term coherency**: the semantic coherency of terms in the same topic
  - Inject a random term into the top terms, and try to identify the injected term

<table>
<thead>
<tr>
<th>Method</th>
<th>Relation Accuracy</th>
<th>Term Coherency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DBLP</td>
<td>SP</td>
</tr>
<tr>
<td>HPAM</td>
<td>0.109</td>
<td>0.160</td>
</tr>
<tr>
<td>HLDA</td>
<td>0.272</td>
<td>0.383</td>
</tr>
<tr>
<td>HClus</td>
<td>0.436</td>
<td>0.240</td>
</tr>
<tr>
<td>NoAC</td>
<td>0.563</td>
<td>0.208</td>
</tr>
<tr>
<td>NoLE</td>
<td>0.645</td>
<td>0.240</td>
</tr>
<tr>
<td>TaxonGen</td>
<td><strong>0.775</strong></td>
<td><strong>0.520</strong></td>
</tr>
</tbody>
</table>

- **HPAM**: Hierarchical Pachinko Allocation Model
- **HLDA**: Hierarchical Latent Dirichlet Allocation
- **HClus**: Hierarchical Spherical k-means clustering
- **NoAC**: A Variant without Adaptive Clustering
- **NoLE**: A Variant without Local Embedding
Davies-Bouldin index: Measure how well clusters are separated in the semantic space

\[ DB(C) = \frac{1}{K} \sum_{i=1}^{K} \max_{i \neq j} \left\{ \frac{\Delta(C_i) + \Delta(C_j)}{\delta(C_i, C_j)} \right\} \]

- Compare intra-cluster against inter-cluster similarity
- The smaller, the better separated of the clusters

(a) DB index on DBLP.

(b) DB index on SP.
Outline

- Taxonomy Basics
- Clustering-based Taxonomy Construction
- TaxoGen: Topic Taxonomy Construction by Adaptive Term Embedding and Clustering
- Instance-based Taxonomy Construction
- HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion
- Summary
Instance-based Taxonomy Construction: Overview

- Decompose taxonomy construction into multiple subtasks

Input data:
- Text corpus
- OR/AND
- Term List

Hypernymy Detection

Extracted Pairs
- <panda, mammal>
- <lizard, reptile>
- <reptile, vertebrate>
- <dog, mammal>
- <cat, mammal>
- <mammal, vertebrate>
- ...

Hypernymy Organization

- End-to-end approach
- Pattern-based approach
- Supervised approach
- Simple pruning heuristics
- Graph-based approach
Hypernymy Detection

- Pattern-based approach: use patterns to extract hypernym-hyponym relations from raw text
- Lexical-syntactic pattern [Hearst’92] [Kozareva and Hovy’10], [Luu et al.’14]
- Generalized pattern:
  - Star patterns [Navigli and Velardi’10]
  - SOL pattern [Nakashole et al.’12]
  - Meta pattern [Jiang et al.’17]

Use bootstrapping to acquire more patterns

<table>
<thead>
<tr>
<th>Extracted Pairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;panda, mammal&gt;</td>
</tr>
<tr>
<td>&lt;dog, mammal&gt;</td>
</tr>
<tr>
<td>&lt;mammal, animal&gt;</td>
</tr>
</tbody>
</table>
Hypernymy Detection

- Supervised approach: train a classifier to predict whether two terms in vocabulary hold hypernymy relation

- Leverage multiple features:
  - Term embedding: [Fu et al.’14] [Yu et al.15] [Luu et al.’16] [Weeds et al.’16]
  - Dependency path: [Snow et al.’04] [Snow et al.’06] [Shwartz et al.’16] [Mao et al.’18]

### Annotated Pairs
- <panda, mammal>
- <dog, mammal>
- <dog, pet>
- <cat, pet>
- <mammal, animal>
  ...

### Unlabeled Pairs

<table>
<thead>
<tr>
<th>Unlabeled Pairs</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;tiger, mammal&gt;</td>
<td>True</td>
</tr>
<tr>
<td>&lt;fish, mammal&gt;</td>
<td>False</td>
</tr>
</tbody>
</table>
Simple pruning heuristics:

- Remove cycle [Kozareva and Hovy’10] [Faralli et al.’15]
- Retain longest-path [Kozareva and Hovy’10]

Figure credits to [Kozareva and Hovy’10]
Graph-based approach:
- Maximum Spanning Tree [Paola et al.'13] [Bansal et al.'14] [Zhang et al.'16]
- Apply Chu-Liu/Edmonds’s algorithm for finding optimal branching

Figure credits to [Paola et al.'13]

Noisy Graph
Trimmed Graph with Edge Weights
Induced DAG
Hypernymy Organization

- Graph-based approach:
  - Maximum Spanning Tree [Paola et al.’13] [Bansal et al.’14] [Zhang et al.’16]

Figure credits to [Gupta et al.’17]
End-to-End Approach for Taxonomy Construction

- Use structured learning to model taxonomy as a factor graph and use learned relation probability to weight hypernymy relation [Bansal et al.'14]
- Use reinforcement learning to learn a policy which attaches terms in vocabulary into taxonomy tree one-by-one [Mao et al.'18]
- Both requires tree-structured training data
Limitations of Existing Methods

- Limitations: Build a corpus-agnostic, task-agnostic taxonomy with mainly is-A relation

- Inflexible semantics: cannot model flexible edge semantics (e.g., “country-state-city”)

- Limited applicability: cannot fit user-specific application tasks
Outline

- Taxonomy Basics
- Clustering-based Taxonomy Construction
- TaxoGen: Topic Taxonomy Construction by Adaptive Term Embedding and Clustering
- Instance-based Taxonomy Construction
- HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion
- Summary
HiExpan: User/Task-Guided Taxonomy Construction

- **Input:** A user provides:
  - a domain-specific corpus, and
  - a seed taxonomy as task guidance

- **Model outputs:**
  - A corpus-dependent taxonomy tailored for user’s task

- **Distinction:** Task-guided taxonomy construction
  - Corpus-dependent
  - Leverage user’s seed guidance

---

Shen, Jiaming, Zeqiu Wu, Dongming Lei, Chao Zhang, Xiang Ren, Michelle Vanni, Brian M. Sadler and Jiawei Han. “HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion.” KDD (2018)
The HiExpan core idea: View all children under each taxonomy node forming a *coherent set* and build the taxonomy by expanding all these sets

- Use set expansion algorithm to expand all sets
- Recursively expand the sets in a top-down fashion

*Width expansion:* The width of taxonomy tree increases (i.e., expanded)
Width Expansion: Key Techniques

- Preprocessing: Key term extraction
  - Use AutoPhrase to generate a key term list and identify their in-corpus occurrences
  - Apply POS tagger to obtain the POS tag sequence of each key term occurrence
  - Retain those labeled with a noun POS tag (e.g., “NN”, “NNS”, “NNP”) over 75% time
  - Ensure high recall: Generate candidate key terms, ensure broad selection

- Width Expansion: Use SetExpan + term embedding + better leverage entity type
  - {“U.S.”, “China”} → {“Canada”, “Mexico” ...}, put under parent “Root”
  - Also, {“California”, “Illinois”} → {“Michigan”, “Texas” ...}, put under “U.S.”
  - Features: (i) skip pattern, (ii) term embedding, and (iii) Probase type
  - Similarity measure: Siblings should have similar contexts, embeddings, and types
SetExpan: An Iteratively Set Expansion Algorithm

- SetExpan [1] iteratively expands the set via context feature selection and rank ensemble.
- In HiExpan, we incorporate additional embedding and better type features into SetExpan.

How to Dig Deeper? Cold-Start with Empty Initial Seed Set

- Newly-added nodes in taxonomy tree do not have any child node
  - How to acquire a target node’s initial children?
- Depth Expansion
  - Based on US (California, Illinois, ...), find Canada (Ontario, Quebec, ...), Mexico (...)
  - Based on term embedding and embedding vector similarity

How to find these initial children node?
**Depth Expan: Weakly-Supervised Relation Extraction**

- Previous research [1,2] shows that the embedding offset can indicate semantic relation
- Use REPEL [2] to incorporate user supervision for embedding learning

After learning embedding, we expect to have:

\[
v(“U.S”) - v(“California”) \approx v(“Canada”) - v(“Ontario”)
\]

We select top-ranked candidate nodes as initial children node:

\[
sim_{par}(e_t, e_x) = \cos(v(e_t) - v(e_x), \frac{1}{|E|} \sum_{(e_p, e_c)} v(e_p) - v(e_c))
\]

A target node without children

A candidate node to be scored

A set of reference edges

[1] Mikolov et al., Distributed Representations of Words and Phrases and their Compositionality, NIPS 2013
[2] Qu et al., Weakly-supervised Ranking Extraction by Pattern-enhanced Embedding Learning, WWW 2018
HiExpan: Conflict Resolution

- Iteratively applying width & depth Expansion, fully expand the hierarchical tree
- Conflict Resolution
  - Ex. What about “Texas” was selected in both U.S. and Mexico?
  - Calculate each node’s confidence score: “Texas” has a stronger relation with other states in U.S. than those in Mexico, select the “Texas” node under “U.S.”
  - Let “Mexico” remember it to prevent from the node being added back later
HiExpan: Taxonomy Global Optimization

- Width & depth expansions leverage only local information
- Possible conflicts: A term may appear in multiple positions in the tree
- An expanded node may not be under its best “parent” node
- Conflict resolution after each iteration
  - Find the best position for each conflict
  - Delete all other positions
- Global taxonomy adjustment at the end of iteratively expansion process
  - Re-allocate children nodes into parent “classes”
HiExpan: Experiments

- Testing on three datasets from different domains
  - Wiki-location, DBLP-subarea, PubMed-CVD
  - Manually label the quality of constructed taxonomies

- Datasets: (statistics)

- Evaluation Metrics:
  - Ancestor-Precision/Recall/F1: Measures correctly predicted ancestral relations
  - Edge-Precision/Recall/F1: Measures correctly predicted parenthood relations

### Table 1: Datasets statistics

<table>
<thead>
<tr>
<th>Dataset</th>
<th>File Size</th>
<th># Sentences</th>
<th># Entities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wiki</td>
<td>1.02GB</td>
<td>1.50M</td>
<td>41.2K</td>
</tr>
<tr>
<td>DBLP</td>
<td>520MB</td>
<td>1.10M</td>
<td>17.1K</td>
</tr>
<tr>
<td>PubMed-CVD</td>
<td>1.60GB</td>
<td>4.48M</td>
<td>36.1K</td>
</tr>
</tbody>
</table>

### Results

<table>
<thead>
<tr>
<th>Methods</th>
<th>Wiki</th>
<th>DBLP</th>
<th>PubMed-CVD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$P_a$</td>
<td>$R_a$</td>
<td>$F_{1a}$</td>
</tr>
<tr>
<td></td>
<td>$P_e$</td>
<td>$R_e$</td>
<td>$F_{1e}$</td>
</tr>
</tbody>
</table>

- **HSetExpan**: hierarchical SetExpan method
- **NoREPEL**: a variant without REFEL module
- **NoGTO**: a variant without final global tree optimization module
- **HiExpan**: full version of the HiExpan framework
HiExpan Case Study (1): Wiki-Location Taxonomy

The diagram illustrates a hierarchy of locations, starting from the root node and branching out to various countries and cities. The tree structure shows the relationships between different regions, with countries as parent nodes and cities as child nodes. For example, the United States has states as children, and the United Kingdom has cities and locations as children. This type of taxonomy can be used to organize and navigate through a large collection of data related to locations.
HiExpan Case Study (2): DBLP-CS Taxonomy

- root
  - data_mining
    - machine_learning
      - decision_trees
      - neural_networks
      - association_rule_mining
      - text_mining
      - outlier_detection
      - named_entity_recognition
      - information_extraction
      - machine_translation
    - support_vector_machines
    - natural_language_processing
      - information_retrieval
      - wireless_networks
      - pattern_recognition
      - artificial_intelligence
    - image_processing
      - signal_processing
      - computational_biology
      - data_analysis
      - medical_imaging
  - machine_learning
    - data_mining
    - natural_language_processing
      - information_retrieval
      - wireless_networks
      - pattern_recognition
      - artificial_intelligence
    - image_processing
      - signal_processing
      - computational_biology
      - data_analysis
      - medical_imaging
  - named_entity_recognition
  - information_extraction
  - machine_translation
  - question_answering
  - text_summarization
  - word Sense Disambiguation
  - text_classification
  - recommendation_systems
  - knowledge_discovery
  - knowledge_representation
  - computational_linguistics
  - image Enhancement
  - image_compression
  - texture_classification
  - skin_detection
  - image_segmentation
  - pose_estimation

natural_language_processing
data_mining
machine_learning
root
outlier_detection
text_mining
association_rule_mining
image_compression
image_enhancement
image_segmentation
skin_detection
texture_classification
word Sense Disambiguation
knowledge_discovery
knowledge_representation
computational_linguistics
image Enhancement
image_compression
texture_classification
skin_detection
image_segmentation
pose_estimation
Before and After Taxonomy Global Optimization

- Taxonomy global optimization module can improve the quality of constructed taxonomy

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Entity</th>
<th>NoGTO</th>
<th>HiExpan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wiki</td>
<td>London</td>
<td>Australia</td>
<td>England</td>
</tr>
<tr>
<td></td>
<td>Chiba</td>
<td>China</td>
<td>Japan</td>
</tr>
<tr>
<td></td>
<td>Molise</td>
<td>Frances</td>
<td>Italy</td>
</tr>
<tr>
<td></td>
<td>New_South_Wales</td>
<td>England</td>
<td>Australia</td>
</tr>
<tr>
<td></td>
<td>Shropshire</td>
<td>Scotland</td>
<td>England</td>
</tr>
<tr>
<td>DBLP</td>
<td>unsupervised_learning</td>
<td>data_mining</td>
<td>machine_learning</td>
</tr>
<tr>
<td></td>
<td>social_network_analysis</td>
<td>natural_language_processing</td>
<td>data_mining</td>
</tr>
<tr>
<td></td>
<td>multi-label_classification</td>
<td>information_retrieval</td>
<td>machine_learning</td>
</tr>
<tr>
<td></td>
<td>pseudo-relevance_feedback</td>
<td>computational_biology</td>
<td>information_retrieval</td>
</tr>
<tr>
<td></td>
<td>function_approximate</td>
<td>data_analysis</td>
<td>machine_learning</td>
</tr>
</tbody>
</table>
Outline

- Taxonomy Basics
- Clustering-based Taxonomy Construction
- TaxoGen: Topic Taxonomy Construction by Adaptive Term Embedding and Clustering
- Instance-based Taxonomy Construction
- HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion
- Summary
Summary

- Automated taxonomy construction is a critical task
  - Dramatically reduce human/expert’s effects
  - Benefit a wide range of downstream applications
- Clustering-based Taxonomy Construction
  - Hierarchical Topic Model, Hierarchical Clustering
  - TaxoGen: Topic Taxonomy Construction by Adaptive Term Embedding and Clustering
- Instanced-based Taxonomy Construction:
  - Hypernymy Detection + Hypernymy Organization, End-to-End Approach
  - HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion
References: Clustering-based Taxonomy Construction


- Wei Li and Andrew D McCallum. 2006. Pachinko allocation: DAG-structured mixture models of topic correlations. In ICML.


- Chi Wang, Marina Danilevsky, Nihit Desai, Yinan Zhang, Phuong Nguyen, rivikrama Taula, and Jiawei Han. 2013. A phrase mining framework for recursive construction of a topical hierarchy. In KDD.

- Chi Wang, Jialu Liu, Nihit Desai, Marina Danilevsky, and Jiawei Han. 2013. Constructing topical hierarchies in heterogeneous information networks. ICDM (2013).

- Xueqing Liu, Yangqiu Song, Shixia Liu, and Haixun Wang. 2012. Automatic taxonomy construction from keywords. In KDD.


- Zhang, Chao, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian M. Sadler, Michelle Vanni and Jiawei Han. “TaxoGen : Unsupervised Topic Taxonomy Construction by Adaptive Term Embedding and Clustering.” KDD 2018
References: Instance-based Taxonomy Construction I

❑ Anh Tuan Luu, Jung Jae Kim, and See-Kiong Ng. 2014. Taxonomy Construction Using Syntactic Contextual Evidence. In EMNLP.
❑ Simone Paolo Ponzetto and Roberto Navigli. 2009. Large-Scale Taxonomy Mapping for Restructuring and Integrating Wikipedia. In IJCAI.
❑ Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2004. Learning Syntactic Patterns for Automatic Hypernym Discovery. In NIPS.
❑ Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2006. Semantic Taxonomy Induction from Heterogenous Evidence. In ACL.
❑ Ndapandula Nakashole, Gerhard Weikum, and Fabian M. Suchanek. 2012. PATTY: A Taxonomy of Relational Patterns with Semantic Types. In EMNLP-CoNLL.
❑ Roberto Navigli and Paola Velardi. 2010. Learning Word-Class Lattices for Definition and Hypernym Extraction. In ACL.
❑ Luis Espinosa Anke, José Camacho-Collados, Claudio Delli Bovi, and Horacio Saggion. 2016. Supervised Distributional Hypernym Discovery via Domain Adaptation. In EMNLP.
❑ Mao, Yuning, Xiang Ren, Jiaming Shen, Xiaotao Gu and Jiawei Han. “End-to-End Reinforcement Learning for Automatic Taxonomy Induction.” ACL (2018)
References: Instance-based Taxonomy Construction II

- Shen, Jiaming, Zeqiu Wu, Dongming Lei, Chao Zhang, Xiang Ren, Michelle Vanni, Brian M. Sadler and Jiawei Han. “HiExpan : Task-Guided Taxonomy Construction by Hierarchical Tree Expansion.” KDD (2018).
- Roberto Navigli, Paola Velardi and Stefano Faralli , “A Graph-based Algorithm for Inducing Lexical Taxonomies from Scratch ”. In IJCAI, 2011
Automatic Synonym Discovery with Knowledge Bases

- Meng Qu, Xiang Ren and Jiawei Han, "Automatic Synonym Discovery with Knowledge Bases", KDD'17

- Entity synonyms widely exist in human languages

<table>
<thead>
<tr>
<th>Entity</th>
<th>Synonyms</th>
</tr>
</thead>
<tbody>
<tr>
<td>United States</td>
<td>the United States, USA, America</td>
</tr>
<tr>
<td>Washington State</td>
<td>Washington State, the State of Washington</td>
</tr>
<tr>
<td>Aspirin</td>
<td>Aspirin, Acetylsalicylic Acid</td>
</tr>
</tbody>
</table>

- Synonym discovery: an important task, benefiting many applications
Why Automatic Synonym Discovery with Knowledge Bases?

- Limitations of existing approaches
  - Suffer from ambiguity problem: One name string may refer to multiple entities
  - Rely on careful seed selections by domain experts: high costs and non scalable
- Proposed approach: Automatic synonym discovery with knowledge bases

- Collect existing entity synonyms in knowledge bases as distant supervision
- Identify missing synonyms for knowledge base entities by leveraging their existing synonyms to disambiguate
Key Task and Existing Methods: 1. Distribution-Based

- Key Task: Given two target strings, predict whether they are *synonymous*

<table>
<thead>
<tr>
<th>Target Strings</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>USA, United States</td>
<td>😊</td>
</tr>
<tr>
<td>USA, Canada</td>
<td>😞</td>
</tr>
</tbody>
</table>

- Existing Methods: *Distributional* based vs. *Pattern* based

  - *Distributional* based methods
    - Consider *global* corpus-level statistics
    - Represent strings with their distributional features
    - Make predictions based on the representations
  
  - Strong Points: High recall; don’t require target strings to co-occur in any sentences
  
  - Weak Points: Low precision
    - Some non-synonymous strings can share similar distributional features
      - E.g., USA and Canada

Distributional Based Approaches

<table>
<thead>
<tr>
<th></th>
<th>Country</th>
<th>State</th>
<th>The</th>
</tr>
</thead>
<tbody>
<tr>
<td>America</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>USA</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
Existing Methods: 2. Pattern-Based

- Idea:
  - Consider *local* contexts
  - Extract textual patterns from local contexts
  - Make predictions based on the patterns
- Strong Points: Good interpretability
- Weak Points:
  - Low recall
  - Many synonymous strings will not co-occur in any sentences
  - Only 45% synonymous strings will co-occur in some sentences
  - Some patterns are not so reliable
- Prior to 1970 the *group was known as* curtain time USA
- The *event is called* Christmas town USA
Our Idea: Integration of Two Methods

- Comparison of the distributional methods and pattern methods
  - They are *complementary*

<table>
<thead>
<tr>
<th>Methods</th>
<th>Precision</th>
<th>Recall</th>
<th>Interpretability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributional</td>
<td>Bad</td>
<td>Good</td>
<td>Bad</td>
</tr>
<tr>
<td>Pattern-based</td>
<td>Not Good</td>
<td>Bad</td>
<td>Good</td>
</tr>
</tbody>
</table>

- Idea:
  - Integrate the *distributional* methods and the *pattern* methods
  - Predict synonyms from both the *global* statistics and the *local* contexts
Proposed Framework

Seed Collection

- Text Corpus
- Synonym Seeds

Model Learning

- Distributional Module
- Pattern Module

Inference

- Query Entity
- Discovered Synonyms

- High-Potential Candidates

- Distributional
- Pattern

Seed Collection

Model Learning

- Two modules: Distributional module and pattern module
- String embeddings:
  - The two modules share the same string embeddings
  - They treat the embeddings as features, and in turn update the embeddings

Inference
Distributional Module: Unsupervised Part

- **Goal:**
  - Preserve the semantic meanings of strings into the embeddings
  - Learn a distributional score function for synonym prediction

- **Unsupervised Part:**
  - Construct a string co-occurrence network of strings
  - Preserve the encoded semantic meanings into the embeddings

**Observation 3.1 (Co-occurrence Observation):**
1. If two strings have similar semantic meanings, then they are more likely to co-occur with each other.
2. If a string tends to appear in the context of another one, then they tend to co-occur frequently.

**Conditional Probability**

$$p(u|v) = \frac{\exp(x_u^T x_v + x_u^T c_v)}{Z},$$

**Objective Function**

$$L_C = \sum_{u,v \in V} w_{u,v} \log p(u|v),$$

- The semantic meaning of $v$
- What strings are likely to co-occur with $v$

Examples:
- “Data Mining” and “Text Mining”
- “USA” and “capital”
Distributional Module: Supervised Part

- Supervised Part:
  - Treat string embeddings as features
  - Treat the synonym seeds as supervision
  - Learn a distributional score function for synonym prediction

The D score of a string pair \((u,v)\)

\[ \text{Score}_D(u,v) = x_u W_D x_v^T, \]

Objective Function

\[ L_S = \sum_{(u,v) \in S_{\text{seed}}} \sum_{v' \in V} \min(1, \text{Score}_D(u,v) - \text{Score}_D(u,v')), \]
Pattern Module

- **Goal:**
  - Learn a pattern classifier to predict synonyms from local contexts

- **Method:**
  - For the target strings, extract sentences mentioning both of them
  - Extract a pattern from each sentence
  - Extract some lexical and syntactic features from each pattern
  - Classify the patterns based on the features

---

**Objective Function**

\[ O_P = \sum_{\text{pat} \in S_{\text{pat}}} \log P(y_{\text{pat}} | \text{pat}) \]

**The P score of a string pair \((u,v)\)**

\[ \text{Score}_P(u,v) = \frac{\sum_{\text{pat} \in S_{\text{pat}}(u,v)} P(y_{\text{pat}} = 1 | \text{pat})}{|S_{\text{pat}}(u,v)|} \]
Optimization and Inference

- Optimization:
  - Objective function: \( L_C + L_S + O_P \)
  - Modeling Co-occurrence Network
  - Modeling the Seeds
  - Modeling the Patterns

- Optimization:
  - Alternatively sample from training examples from the three parts
  - Use the SGD algorithm for optimization

- Inference:
  - We use both the distributional score and the pattern score to re-rank the candidate strings

\[
Score(e, u) = \sum_{s \in S_{syn}(e)} \{ Score_D(s, u) + \lambda Score_P(s, u) \}.
\]
Experiment Setting

- Task: Predict synonyms for knowledge base entities
  - Warm start setting: Assume 50% entity synonyms are known, and aim to predict the rest 50%
  - Cold start setting: Assume only the original entity names are known, and aim to predict the rest

- Evaluation metrics: P@K, R@K, F1@K.

- Datasets:
  - Wiki + Freebase
  - PubMed + UMLS
  - NYT + Freebase

- Lambda: 0.1

---

Table 1: Statistics of the Datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Wiki</th>
<th>PubMed</th>
<th>NYT</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Documents</td>
<td>100,000</td>
<td>1,554,433</td>
<td>118,664</td>
</tr>
<tr>
<td>#Sentences</td>
<td>6,839,331</td>
<td>15,051,203</td>
<td>3,002,123</td>
</tr>
<tr>
<td>#Strings in Vocab</td>
<td>277,635</td>
<td>357,985</td>
<td>115,680</td>
</tr>
<tr>
<td>#Training Entities</td>
<td>4,047</td>
<td>9,298</td>
<td>1,219</td>
</tr>
<tr>
<td>#Test Entities (Warm)</td>
<td>256</td>
<td>250</td>
<td>79</td>
</tr>
<tr>
<td>#Test Entities (Cold)</td>
<td>175</td>
<td>150</td>
<td>72</td>
</tr>
</tbody>
</table>
Experimental Results

**Table 2: Quantitative results on the warm-start setting.**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Wiki + Freebase</th>
<th>PubMed + UMIS</th>
<th>NYT + Freebase</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P@1 R@1 F1@1 P@5 R@5 F1@5</td>
<td>P@1 R@1 F1@1 P@5 R@5 F1@5</td>
<td>P@1 R@1 F1@1 P@5 R@5 F1@5</td>
</tr>
<tr>
<td>Patty</td>
<td>0.102 0.075 0.086 0.049 0.167 0.067</td>
<td>0.352 0.107 0.164 0.164 0.248 0.197</td>
<td>0.101 0.081 0.090 0.038 0.141 0.060</td>
</tr>
<tr>
<td>SVM</td>
<td>0.508 0.374 0.431 0.273 0.638 0.382</td>
<td>0.696 0.211 0.324 0.349 0.515 0.416</td>
<td>0.481 0.384 0.427 0.248 0.616 0.354</td>
</tr>
<tr>
<td>word2vec</td>
<td>0.387 0.284 0.328 0.247 0.621 0.533</td>
<td>0.784 0.238 0.365 0.464 0.659 0.545</td>
<td>0.367 0.293 0.326 0.216 0.596 0.317</td>
</tr>
<tr>
<td>GloVe</td>
<td>0.254 0.187 0.215 0.104 0.316 0.156</td>
<td>0.536 0.163 0.250 0.279 0.417 0.334</td>
<td>0.203 0.162 0.180 0.084 0.283 0.130</td>
</tr>
<tr>
<td>PTR</td>
<td>0.445 0.324 0.378 0.252 0.642 0.357</td>
<td>0.690 0.243 0.397 0.476 0.674 0.559</td>
<td>0.456 0.364 0.405 0.223 0.606 0.357</td>
</tr>
<tr>
<td>RKPM</td>
<td>0.500 0.368 0.424 0.302 0.681 0.418</td>
<td>0.804 0.244 0.374 0.480 0.677 0.562</td>
<td>0.506 0.404 0.449 0.302 0.707 0.423</td>
</tr>
<tr>
<td>DPE-NoP</td>
<td>0.641 0.471 0.534 0.414 0.790 0.543</td>
<td>0.816 0.247 0.379 0.332 0.733 0.617</td>
<td>0.532 0.424 0.472 0.305 0.687 0.422</td>
</tr>
<tr>
<td>DPE</td>
<td>0.727 0.534 0.616 0.465 0.816 0.595</td>
<td>0.872 0.265 0.406 0.349 0.753 0.636</td>
<td>0.570 0.455 0.506 0.366 0.788 0.580</td>
</tr>
<tr>
<td>DPE-NoP</td>
<td>0.641 0.471 0.534 0.414 0.790 0.543</td>
<td>0.816 0.247 0.379 0.332 0.733 0.617</td>
<td>0.532 0.424 0.472 0.305 0.687 0.422</td>
</tr>
<tr>
<td>DPE-TwoStep</td>
<td>0.684 0.503 0.580 0.417 0.782 0.544</td>
<td>0.836 0.254 0.390 0.358 0.748 0.628</td>
<td>0.557 0.444 0.494 0.344 0.768 0.475</td>
</tr>
<tr>
<td>DPE</td>
<td>0.727 0.534 0.616 0.465 0.816 0.595</td>
<td>0.872 0.265 0.406 0.349 0.753 0.636</td>
<td>0.570 0.455 0.506 0.366 0.788 0.580</td>
</tr>
</tbody>
</table>

**Table 3: Quantitative results on the cold-start setting.**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Wiki + Freebase</th>
<th>PubMed + UMIS</th>
<th>NYT + Freebase</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P@1 R@1 F1@1 P@5 R@5 F1@5</td>
<td>P@1 R@1 F1@1 P@5 R@5 F1@5</td>
<td>P@1 R@1 F1@1 P@5 R@5 F1@5</td>
</tr>
<tr>
<td>Patty</td>
<td>0.131 0.056 0.078 0.065 0.136 0.088</td>
<td>0.413 0.064 0.111 0.191 0.148 0.167</td>
<td>0.125 0.054 0.075 0.062 0.132 0.084</td>
</tr>
<tr>
<td>SVM</td>
<td>0.371 0.158 0.222 0.150 0.311 0.202</td>
<td>0.707 0.110 0.193 0.381 0.297 0.334</td>
<td>0.347 0.150 0.209 0.165 0.347 0.224</td>
</tr>
<tr>
<td>word2vec</td>
<td>0.411 0.175 0.245 0.196 0.401 0.263</td>
<td>0.627 0.098 0.170 0.408 0.318 0.357</td>
<td>0.361 0.156 0.218 0.151 0.317 0.205</td>
</tr>
<tr>
<td>GloVe</td>
<td>0.251 0.107 0.150 0.105 0.221 0.142</td>
<td>0.480 0.075 0.130 0.264 0.206 0.231</td>
<td>0.181 0.078 0.109 0.084 0.189 0.115</td>
</tr>
<tr>
<td>PTE</td>
<td>0.474 0.202 0.283 0.227 0.457 0.303</td>
<td>0.647 0.101 0.178 0.389 0.303 0.361</td>
<td>0.405 0.174 0.245 0.166 0.347 0.228</td>
</tr>
<tr>
<td>RKPM</td>
<td>0.480 0.204 0.286 0.227 0.455 0.303</td>
<td>0.709 0.109 0.189 0.447 0.348 0.391</td>
<td>0.403 0.186 0.255 0.170 0.353 0.229</td>
</tr>
<tr>
<td>DPE-NoP</td>
<td>0.491 0.209 0.293 0.246 0.491 0.328</td>
<td>0.709 0.109 0.189 0.456 0.355 0.399</td>
<td>0.417 0.180 0.251 0.180 0.371 0.242</td>
</tr>
<tr>
<td>DPE</td>
<td>0.646 0.275 0.386 0.302 0.574 0.396</td>
<td>0.753 0.117 0.203 0.300 0.369 0.438</td>
<td>0.486 0.201 0.284 0.207 0.400 0.273</td>
</tr>
<tr>
<td>DPE-NoP</td>
<td>0.641 0.471 0.534 0.414 0.790 0.543</td>
<td>0.816 0.247 0.379 0.332 0.733 0.617</td>
<td>0.532 0.424 0.472 0.305 0.687 0.422</td>
</tr>
<tr>
<td>DPE-TwoStep</td>
<td>0.684 0.503 0.580 0.417 0.782 0.544</td>
<td>0.836 0.254 0.390 0.358 0.748 0.628</td>
<td>0.557 0.444 0.494 0.344 0.768 0.475</td>
</tr>
<tr>
<td>DPE</td>
<td>0.727 0.534 0.616 0.465 0.816 0.595</td>
<td>0.872 0.265 0.406 0.349 0.753 0.636</td>
<td>0.570 0.455 0.506 0.366 0.788 0.580</td>
</tr>
</tbody>
</table>

DPE outperforms Patty: we leverage distributional information, which may help improve the precision and recall.

DPE outperforms the baselines based on distributional information: we leverage the co-occurrence observation to better capture the semantic meanings of strings, and we leverage local patterns to improve the performances.

DPE outperforms DPE-NoP: the pattern information can indeed help improve the performances.
Study on Precision

- Limitation of the above setting: The precision is low
- We care more about the F1 score, which highly relies on recall
- To improve the recall, a string will be highly ranked if we observe a strong signal from the D module
- In some cases, we care more about the precision
- To improve the precision, a string will be highly ranked if we observe strong signals from both the D module and the P module

<table>
<thead>
<tr>
<th>String</th>
<th>D score</th>
<th>P score</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.95</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>0.9</td>
<td>0.1</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>0.8</td>
<td>0.8</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>String</th>
<th>D score</th>
<th>P score</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>0.8</td>
<td>0.8</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>0.9</td>
<td>0.1</td>
<td>2</td>
</tr>
<tr>
<td>A</td>
<td>0.95</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>
Performance on Precision

- We study the precision on the Wiki-Freebase dataset (warm-start setting) with the new ranking method
  - P@10: 100%   P@50: 88%   P@100: 89%

- Example output:

<table>
<thead>
<tr>
<th>Rank</th>
<th>Entity</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Anatolia</td>
<td>Asia minor</td>
</tr>
<tr>
<td>2</td>
<td>e-mail</td>
<td>email</td>
</tr>
<tr>
<td>3</td>
<td>Daoism</td>
<td>Taoism</td>
</tr>
<tr>
<td>4</td>
<td>Byzantine Empire</td>
<td>Eastern Roman Empire</td>
</tr>
<tr>
<td>5</td>
<td>East Germany</td>
<td>GDR</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rank</th>
<th>Entity</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>96</td>
<td>Eyeglasses</td>
<td>glasses</td>
</tr>
<tr>
<td>97</td>
<td>BSE</td>
<td>mad cow disease</td>
</tr>
<tr>
<td>98</td>
<td>Saint Mary</td>
<td>Saint John</td>
</tr>
<tr>
<td>99</td>
<td>Cocoa</td>
<td>Cacao</td>
</tr>
<tr>
<td>100</td>
<td>British Open</td>
<td>the Open Championship</td>
</tr>
</tbody>
</table>

Our method can also achieve very high precision by adjusting the ranking strategy
Case Study

- Example output for query entities:

<table>
<thead>
<tr>
<th>Entity</th>
<th>US dollar</th>
<th>World War II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>DPE-NoP</td>
<td>DPE</td>
</tr>
<tr>
<td>US Dollars</td>
<td>US dollar</td>
<td>Second World War</td>
</tr>
<tr>
<td>U.S. dollars</td>
<td>US dollars</td>
<td>World War Two</td>
</tr>
<tr>
<td>Euros</td>
<td>U.S. dollars</td>
<td>World War One</td>
</tr>
<tr>
<td>U.S. dollar</td>
<td>U.S. $</td>
<td>WW I</td>
</tr>
<tr>
<td>RMB</td>
<td>Euros</td>
<td>world wars</td>
</tr>
</tbody>
</table>

- Top ranked patterns for the synonym relation:

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Corresponding Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-,NN,subj) (-lrb- JJ, amod) (known, VBN, acl) (-,NN, nmod)</td>
<td>... <em>Olympia</em> (commonly known as <em>L’Olympia</em>) is a music hall ...</td>
</tr>
<tr>
<td>(-,NN, dobj) (-,NN, appos)</td>
<td>... , many hippies used <em>cannabis</em> (marijuana), considering it ...</td>
</tr>
<tr>
<td>(-,NNP, subj) (known, VBN, acl) (-,NN, nmod)</td>
<td>... <em>BSE</em>, commonly known as &quot;mad cow disease&quot;, is a ...</td>
</tr>
</tbody>
</table>
Conclusions and Future Work

- Conclusions
  - The distributional and the pattern information can well complement each other
  - They can mutually collaborate to achieve high precision and recall

- Future work
  - Apply the idea to some information extraction problems
  - Propose more effective methods to integrate the two signals
  - Framework: let them mutually enhance each other
    - D2P: D can clean the facts or the patterns extracted by P, and therefore enhances the P part
    - P2D: P can generate more seeds for D, and therefore enhances the D part