Automated Feature Generation from Structured Knowledge

Weiwei Cheng    Gjergji Kasneci    Thore Graepel

David Stern    Ralf Herbrich

Microsoft Research Cambridge
Most machine learning researches focus on the modeling, while the construction of features is as crucial.

Can we design a mechanism that compactly describes and extracts relevant features?
In our work, we introduce

1. a theoretical framework for constructing semantic features from a given knowledge base;

2. various strategies for incorporating these features into a prediction model.
In our work, we introduce

1. a theoretical framework for constructing semantic features from a given knowledge base;

2. various strategies for incorporating these features into a prediction model.
In our work, we introduce

1. a theoretical framework for constructing semantic features from a given knowledge base;
2. various strategies for incorporating these features into a prediction model.
In our work, we introduce

1. a theoretical framework for constructing semantic features from a given knowledge base;

2. various strategies for incorporating these features into a prediction model.
In our work, we introduce

1. a theoretical framework for constructing semantic features from a given knowledge base;

2. various strategies for incorporating these features into a prediction model.
In our work, we introduce

1. a theoretical framework for constructing semantic features from a given knowledge base;

2. various strategies for incorporating these features into a prediction model.
To query semantic features, we propose

**ESPARQL**

(Extended SPARQL Query Language), an extension of SPARQL with NAGA features.
Retrieve all neighbors of “Albert Einstein” in the knowledge base.

select ?x
where {
  (albert_einstein ?r ?x) union
  (?x ?r albert_einstein)
}

Retrieve all classes “Albert Einstein” belongs to.

select ?x
where {
  albert_einstein type subClass* ?x
}
Exemplary Queries

Find country information based on the user’s IP address.

select ?z
where {
    ?y hasIPAddress userIP .
    userIP belongsToLocation ?z .
    ?z type subClass* country
}

Count the followers of a particular user in Twitter.

select count(?x)
where {
    ?x follows user
}
1. Based on the training corpus and the knowledge base, choose appropriate SPARQL queries to extract semantic information about the entities in the training corpus.
2. Unify the answer sets of the queries to construct the set of semantic features that indexes the dimensions of the effective feature space.
Constructing Feature Vectors

```sql
select ?c
where { h type subClass* ?c }
```

3. The feature vector of an entity, which has the dimensionality of the feature space, can be built by setting those dimensions which correspond to query answers for the entity to 1.

\[
\phi(h) = (1 \ 1 \ 1 \ 0 \ 1 \ 1)
\]

\[
\phi(g) = (1 \ 1 \ 0 \ 1 \ 0 \ 0)
\]
Generalized Linear Bayesian Probit Model

- Given the binary feature vector $\phi(x) = (\phi_1(x), \ldots, \phi_n(x))$, the importance of features is represented by a weight vector $w = (w_1, \ldots, w_n)$, where the belief on $w_i$ is given by a Gaussian distribution with mean $\mu_i$ and $\sigma_i^2$. We have
  \[ p(w) = \prod_i \mathcal{N}(w_i; \mu_i, \sigma_i^2). \]

- The score of instance (i.e., entity) $x$ is $s(x) = \sum_{i=1}^n w_i \phi_i(x)$, and we have
  \[ p(s \mid w, x) = \mathcal{N} \left( s; \sum_{i=1}^n \phi_i(x) \mu_i, \sum_{i=1}^n \phi_i(x) \sigma_i^2 \right). \]

- The output $y$ is modeled as a probit function
  \[ P(y \mid s) = \Phi \left( \frac{ys}{\beta} \right) \]
  with noise variance $\beta^2$, where $\Phi(t) = \int_{-\infty}^{t} \mathcal{N}(v; 0, 1)dv$.  

10/19
Classification of Tweets

Dataset:
Twitter messages were categorized by Amazon Mechanical Turk and then manually verified by researchers in our lab.

<table>
<thead>
<tr>
<th>Category</th>
<th>Tweets</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business / Finance</td>
<td>2,182</td>
<td>9.56%</td>
</tr>
<tr>
<td>Entertainment</td>
<td>4,235</td>
<td>18.56%</td>
</tr>
<tr>
<td>Lifestyle</td>
<td>4,085</td>
<td>17.90%</td>
</tr>
<tr>
<td>Politics</td>
<td>1,199</td>
<td>5.25%</td>
</tr>
<tr>
<td>Science / Environment</td>
<td>789</td>
<td>3.45%</td>
</tr>
<tr>
<td>Sport</td>
<td>1,145</td>
<td>5.01%</td>
</tr>
<tr>
<td>Technology</td>
<td>1,880</td>
<td>8.23%</td>
</tr>
<tr>
<td>World Events</td>
<td>2,122</td>
<td>9.30%</td>
</tr>
<tr>
<td>Other / Miscellaneous</td>
<td>12,838</td>
<td>56.26%</td>
</tr>
</tbody>
</table>

22,816 tweets in total
Ballmer on iPad: “they've sold certainly more than I'd like them to have sold”

Obama blames Bush for all of his misdeeds and then takes credit for the successful war in Iraq http://is.gd/e0iVM (via @PennyStarrDC)
Ballmer on iPad: “they've sold certainly more than I'd like them to have sold”

Obama blames Bush for all of his misdeeds and then takes credit for the successful war in Iraq http://is.gd/e0iVM (via @PennyStarrDC)
In most cases, semantic features are helpful.

Semantic features hardly improve the performance in *Politics* and *Tech*.

Hypernym-based features are not very informative for this task.
Negative log-likelihood

- Strong dependencies among features
- Not all learning models benefit from the additional features
Semantic features using our model

Semantic features using Naïve Bayes

Only bag of words using Naïve Bayes

- Strong dependencies among features
- Not all learning models benefit from the additional features
- Naïve Bayes works even better without semantic features.
Movie Recommendation

Dataset:
**MovieLens**
1,000,206 ratings for 3,900 movies by 6,040 users
Ratings are ordinal scale from 1 to 5. Sparsity: 95.7%.

Conventional collaborative filtering with SVD:

\[
(U_{SVD}, V_{SVD}) := \arg \min_{(U,V)} \sum_{i=1}^{n} \sum_{j=1}^{m} (u_i^T v_j - r_{ij})^2
\]

We bring in the movie type information and the actor information with YAGO.
Semantic features are very helpful during cold start, i.e., for movies that have not been rated frequently.

The effect becomes insignificant when more and more ratings are unveiled.
Summary

• Build the link between semantic technology and machine learning

• Propose framework that compactly describes and extracts relevant features

• Test semantic features in learning (tweet classification, movie recommendation)

• Modularity is a key feature. Choose the knowledge base wisely.

www.chengweiwei.com