Weakly-supervised Text Categorization from Different Sources

Yu Meng
yumeng5@illinois.edu
May 8, 2018
Task Overview

- Text categorization: grouping documents into different categories according to semantic similarity.
- Most conventional supervised/semi-supervised document categorization methods require a large number of documents with labeled categories for training.
  - Hard to be applied in real-world scenarios due to the lack of training data
- How about purely unsupervised learning (clustering)?
  - No training data required – desirable
  - Cannot leverage user guide: especially bad for multidimensional data
  - E.g. a news corpus has 5 topics and 5 locations. If set k=5 for a clustering algorithm, which is the criterion, topics or locations?
- Therefore, weakly-supervised learning
Task Overview

- Weakly-supervised learning: leverage prior knowledge of users to aid text categorization.
  - Higher-level supervision: No labeled documents; only abstract guide like class names
  - Incomplete supervision: Only a small amount of labeled documents available; insufficient to train a good learner
  - Inaccurate supervision: User inputs may contain noises (not considered here)
- We allow user guidance to be of different forms; aim to leverage three different kinds of weak supervision sources:
  - **Labels**: Class label surface names (e.g., “sports”, “arts”, …)
  - **Keywords**: Seed keywords for each class (3~5 keywords per class)
  - **Documents**: Seed documents for each class (10~20 documents per class)
Models

- In this task, we choose to use Convolutional Neural Network (CNN) to perform weakly-supervised text classification, for the following reasons:
  - Unlike SVM or DNN, CNN can exploit meaningful semantic structures (word order) of text data and therefore is the state-of-the-art model for text classification;
  - It is convenient to split the CNN training process into multiple phases, which is required by our task: we first perform a coarse pre-training phase based on weak supervision source, and then a self-learning phase to improve based on current classification results.
Models

- Word2vec embedding

- Document

- Filter size = 3 words
  - Different filter sizes
  - Filter size = 2 words
Methodology Overview

- Word-document-class joint semantic space assumption:
  - Model word, document and class semantic in the same space
  - Handle different kinds of weak supervision sources as input
  - Semantic expansion and similarity analysis in the same space
- Pseudo-document generation:
  - Generate pseudo-documents for each class as training sets
- CNN pre-training and self-training
  - Pre-train CNN with pseudo-documents and apply self-training to improve performance
Joint Semantic Space Assumption

- Joint semantic space:
  - Word2Vec embedding are p-dimensional vectors, \( v_w \in \mathbb{R}^p \)
  - Assume document representation \( d \in \mathbb{R}^p \) as well
  - Vector direction is usually relevant to semantic instead of its magnitude => use normalized vectors (the same reasoning of using cosine similarity)
  - Vectors reside in (p-1) dimensional sphere in \( \mathbb{R}^p \)

- Semantic expansion:
  - Given a class label name (e.g. “sports”), word2vec embedding represents it as one vector
  - However, the semantic of the class is under-represented, because “sports” may also include “athletes”, “NBA”, etc.
  - Therefore, represent a class by a **probability distribution of vector directions** in \( \mathbb{R}^p \)
Joint Semantic Space Assumption
Joint Semantic Space Assumption

- Von Mises–Fisher (vMF) distribution:
  - A probability distribution on the (p-1) dimensional sphere in $R^p$
  - $f(x; \mu, \kappa) = c_p(\kappa)e^{\kappa^T \mu^T x}$ where $c_p(\kappa) = \frac{\Gamma_p(\kappa/2)}{\pi^{p/2}I_{p-1}(\kappa)}$, $I_r(\cdot)$ represents the modified Bessel function of the first kind at order $r$.
  - Parameters interpretation (by analogy with mean and variance in Gaussian distribution):
    - $\mu$ is the semantic centroid; $\kappa$ is the semantic concentration
Joint Semantic Space Assumption

- Semantic expansion – fit a vMF distribution for each class:
  - Given a set of $n$ unit vectors, i.e.,
    
    \[ X = \{x_i \in \mathbb{R}^p \mid x_i \text{ drawn from } f (x; \mu, \kappa), 1 \leq i \leq n \} \]

    we use the maximum likelihood estimates for finding the parameters $\mu^*$ and $\kappa^*$ of the vMF distribution $f (x; \mu, \kappa)$:

    - $\mu^* = \frac{\sum_{i=1}^{n} x_i}{\|\sum_{i=1}^{n} x_i\|}$
    
    \[
    \frac{I_p^{\kappa^*}}{I_p^{\kappa^* - 1}} = \frac{\|\sum_{i=1}^{n} x_i\|}{n}
    \]

    - (Impossible to obtain an analytic solution for $\kappa^*$ because the formula involves an implicit equation; use numerical approximation)
Joint Semantic Space Assumption

- Handling different supervision sources:
  - Source = Labels $w_j$: form a $t$ keyword cluster by finding the top-$t$ words \{w_1, w_2, ..., w_t\} with the most similar word embedding $V = \{v_{w_1}, v_{w_2}, ..., v_{w_t}\}$ with the class label $w_j$; fit a vMF distribution to $V$.
  - Source = Keywords $K_j = \{k_1, k_2, ..., k_n\}$: form a $t$ keyword cluster by finding the top-$t$ words \{w_1, w_2, ..., w_t\} with the most similar word embedding $V = \{v_{w_1}, v_{w_2}, ..., v_{w_t}\}$ with the average word embedding of $K_j$; fit a vMF distribution to $V$.
  - Source = Documents $D_j = \{d_1, d_2, ..., d_n\}$: extract $t$ keywords based on TF-IDF for each class; the remaining procedure becomes the same with Keywords case.

- The joint semantic space assumption provided flexibility for handling different supervision sources!
Pseudo-documents Generation

- Given class $j$’s distribution $f(x; \mu_j, \kappa_j)$, we first sample a document vector $d_i$ from $f(x; \mu_j, \kappa_j)$, then build a keyword vocabulary $V_{d_i}$ for $d_i$ that contains the top-$m$ words with most similar word embedding with $d_i$. 

$d_1$: Music

Arts

$d_2$: Movies
Pseudo-documents Generation

- Given a document vector $d_i$ and its keyword vocabulary $V_{d_i}$, we generate pseudo-document according to the following word distribution:

$$p(w|d_i) = \begin{cases} 
\alpha p_B(w) & w \notin V_{d_i} \\
\alpha p_B(w) + (1 - \alpha) \frac{\exp(d_i^T v_w)}{\sum_{w' \in V_{d_i}} \exp(d_i^T v_{w'})} & w \in V_{d_i}
\end{cases}$$

- where $p_B(w)$ is the background word distribution in the entire corpus
CNN Pre-training

- Generate $k$ pseudo-documents per class and use them to train CNN for initializing its parameters
- A naive way of creating the label for a pseudo-document $D_i^*$ is to treat $D_i^*$ as if it is a given labeled document and use one hot encoding where the class that $D_i^*$ is generated from takes value 1 and all other classes are set to 0
- However, this approach will result in CNN overfitting to pseudo-documents and behaving poorly when classifying real documents
CNN Pre-training

- Design pseudo-labels to prevent overfitting
- Recall that pseudo-document is generated from a mixture of background word distributions weighted by $\alpha$ and the keyword vocabulary weighted by $1 - \alpha$
- Pseudo-label $l_i$ for pseudo-document $D_i^*$ is

$$l_{ij} = \begin{cases} 
(1 - \alpha) + \frac{\alpha}{N} & D_i^* \text{ is generated from class } j \\
\frac{\alpha}{N} & \text{otherwise}
\end{cases}$$
CNN Self-training

- CNN Self-training:
  - Construct a pseudo-label that promotes current high-confidence classification and demotes low-confidence classification.
  
  \[ l_{ij} = \frac{y_{ij}^2 / f_j}{\sum_j y_{ij}^2 / f_j} \], where \( f_j = \sum_i y_{ij} \) (\( y_{ij} \) is the current prediction for \( D_j \))

- Self-training is performed by minimizing KL divergence between current outputs and the pseudo-labels.

\[ loss = \sum_i \sum_j l_{ij} \log \frac{l_{ij}}{y_{ij}} \]
Results

- Datasets:
  - New York Times
  - AG News
  - Yelp

<table>
<thead>
<tr>
<th>Dataset Name</th>
<th>Type</th>
<th>Classes</th>
<th># Documents</th>
<th>Avg Doc Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>AG News</td>
<td>Topic</td>
<td>Politics, Business, Sport, Technology</td>
<td>120000</td>
<td>45</td>
</tr>
<tr>
<td>Yelp</td>
<td>Sentiment</td>
<td>Good, Bad</td>
<td>38000</td>
<td>155</td>
</tr>
</tbody>
</table>
Results

- Effect of pre-training + self-learning:

![Graph showing F1 scores over iterations for different categories: Labels, Keywords, and Docs. The graph shows an increasing trend in F1 scores as iterations progress.](New_York_Times)
Results

- Effect of pre-training + self-learning:
Results

- Effect of pre-training + self-learning:
Results

• Baselines:
  • **IR with TF-IDF:** Class label name/keywords = keyword query; assign the class label according to TF-IDF value of query in each document.
  • **Topic Modeling:** Train LDA model on the entire corpus; assign the class label according to the likelihood of observing a class label name/provided keywords given a document.
  • **Dataless [AAAI’14, AAAI’08]:** Dataless classification takes class names as weak supervision source. It leverages Wikipedia and use explicit semantic analysis to derive vector representations of labels and documents.
Results

- Baselines:
  - **UNEC [SIAM’18]:** Unsupervised neural categorization takes class names as weak supervision source. It categorizes documents by learning the semantics and category attribution of concepts inside the corpus.
  - **CNN [EMNLP’14]:** We use the same CNN model that directly trains on labeled documents as weak supervision source. Then the CNN is used as the classifier to label all documents in the corpus.
  - **PTE [KDD’15]:** Predictive text embedding is a semi-supervised algorithm that utilizes both labeled and unlabeled data to learn text embedding and use logistic regression model as classifier for text categorization.
  - **Pre-training:** Variant of our proposed method but only with pre-training procedure and without self-training procedure.
## Results

- **New York Times:**

<table>
<thead>
<tr>
<th>Methods</th>
<th>Labels</th>
<th>Keywords</th>
<th>Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Macro</td>
<td>Micro</td>
<td>Macro</td>
</tr>
<tr>
<td>IR with TF-IDF</td>
<td>0.319</td>
<td>0.240</td>
<td>0.509</td>
</tr>
<tr>
<td>Topic Modeling</td>
<td>0.301</td>
<td>0.666</td>
<td>0.253</td>
</tr>
<tr>
<td>Dataless</td>
<td>0.484</td>
<td>0.710</td>
<td>-</td>
</tr>
<tr>
<td>UNEC</td>
<td>0.690</td>
<td>0.810</td>
<td>-</td>
</tr>
<tr>
<td>CNN</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PTE</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Pre-training</td>
<td>0.781</td>
<td>0.881</td>
<td>0.676</td>
</tr>
<tr>
<td><strong>This Method</strong></td>
<td><strong>0.844</strong></td>
<td><strong>0.917</strong></td>
<td><strong>0.744</strong></td>
</tr>
</tbody>
</table>

10 documents/class
0.38% of the corpus

Labels as supervision source give the best result.
## Results

### AG News:

10 documents/class
0.03% of the corpus

<table>
<thead>
<tr>
<th>Methods</th>
<th>Labels Macro</th>
<th>Labels Micro</th>
<th>Keywords Macro</th>
<th>Keywords Micro</th>
<th>Documents Macro Avg. (Std.)</th>
<th>Documents Micro Avg. (Std.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IR with TF-IDF</td>
<td>0.187</td>
<td>0.292</td>
<td>0.258</td>
<td>0.333</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Topic Modeling</td>
<td>0.496</td>
<td>0.584</td>
<td>0.723</td>
<td>0.735</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Dataless</td>
<td>0.688</td>
<td>0.699</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>UNEC</td>
<td>0.659</td>
<td>0.668</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CNN</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.766(0.035)</td>
<td>0.769(0.034)</td>
</tr>
<tr>
<td>PTE</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.542(0.029)</td>
<td>0.544(0.031)</td>
</tr>
<tr>
<td>Pre-training</td>
<td>0.679</td>
<td>0.704</td>
<td>0.751</td>
<td>0.758</td>
<td>0.759(0.032)</td>
<td>0.765(0.031)</td>
</tr>
<tr>
<td>This Method</td>
<td><strong>0.829</strong></td>
<td><strong>0.830</strong></td>
<td><strong>0.821</strong></td>
<td><strong>0.823</strong></td>
<td><strong>0.839(0.007)</strong></td>
<td><strong>0.841(0.007)</strong></td>
</tr>
</tbody>
</table>

Documents as supervision source give the best result.
Results

- Yelp:

<table>
<thead>
<tr>
<th>Methods</th>
<th>Labels</th>
<th>Keywords</th>
<th>Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Macro</td>
<td>Micro</td>
<td>Macro</td>
</tr>
<tr>
<td>IR with TF-IDF</td>
<td>0.533</td>
<td>0.548</td>
<td>0.638</td>
</tr>
<tr>
<td>Topic Modeling</td>
<td>0.333</td>
<td>0.500</td>
<td>0.333</td>
</tr>
<tr>
<td>Dataless</td>
<td>0.337</td>
<td>0.500</td>
<td>-</td>
</tr>
<tr>
<td>UNEC</td>
<td>0.602</td>
<td>0.603</td>
<td>-</td>
</tr>
<tr>
<td>CNN</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PTE</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Pre-training</td>
<td>0.506</td>
<td>0.586</td>
<td>0.796</td>
</tr>
<tr>
<td>This Method</td>
<td>0.730</td>
<td>0.731</td>
<td>0.828</td>
</tr>
</tbody>
</table>

20 documents/class
0.1% of the corpus

Keywords as supervision source give the best result
Conclusions and Future Work

- We propose a framework that can leverage different weak supervision sources from user for text categorization.
- Potential applications:
  - Text cube construction
  - Special-domain text retrieval (e.g. dark web)
- Based on the performance across three datasets, we observe that different dataset “favors” different types of weak supervision source:
  - Although **Labels, Keywords, Documents** are different types of weak supervision signals, if leveraged properly, they can all contribute to accurate text categorization
  - In real-world applications, it is not possible to predict which type of supervision source works best for an unseen dataset; therefore, a heterogeneous approach that combines all types of supervision sources will be desirable – future work
Thanks for listening!