A Supervised Machine Learning Algorithm for Research Articles

Leonidas Akritidis, Panayiotis Bozanis

Dept. of Computer & Communication Engineering, University of Thessaly, Greece
Research Article Classification

- We study the problem of categorizing research articles (papers).
- In contrast to regular documents, the research articles have their own features (authors, co-authors, references, publishing journal/conference, etc)
- Therefore, the classification of research articles poses some unique challenges.
Problem Importance

- Allows users search only a specific portion of the document collection.
- Digital libraries organize their content
- Helps users find similar items
- Facilitates the creation of robust search tools
  - Recommendation
  - Query expansion, etc
Existing Approaches (1)

- Keyword extraction algorithms
  - The identify repeated textual patterns and extract the most important terms
  - In the sequel, they employ traditional classification approaches such as kNN.

- Machine learning (ML) approaches
  - Support Vector Machines (SVM)
  - AdaBoost.MH
Existing Approaches (2)

- Citation Analysis Algorithms
  - They exploit linkage information (e.g. multiple articles cited together by a group of papers)
Our Approach

- Employs a set of labels $C$ and a set of pre-classified research papers - training set $\mathcal{T}$.
- It trains a model based on $C$ and $\mathcal{T}$.
- The training phase takes into account the particular features of the problem including:
  - The authors history
  - Co-authorship information
  - Keywords selection
  - The previous publications
Preliminaries

- Our analysis involves four sets:
  - $K$: The set of all keywords ($k \in K$)
  - $A$: The set of all authors ($a \in A$)
  - $J$: The set of all journals ($j \in J$)
  - $C$: The set of all labels ($c \in C$)

- Multiple subsets, i.e.:
  - $K^p$: The subset of keywords of an article $p$.

- And multiple frequency values i.e.:
  - $|P^k|$: The number of articles containing $k$. 
Training Phase

During the training process we build three relevance description vectors (RDV):

\( \mathcal{K} \): Denotes how frequently each keyword \( k \) has been correlated with each field \( c \).

\( \mathcal{A} \): Denotes how frequently each author \( a \) has been correlated with each field \( c \).

\( \mathcal{J} \): Denotes how frequently each journal \( j \) has been correlated with each field \( c \).

Zero freqs are pruned from the vectors.
Model Training (Phase 1)

- In this phase we populate the $\mathcal{K}$ RDV.
- Initially we extract all keywords and labels of the paper (3-4).
- for each pair $(k,c)$ we search within $\mathcal{K}$
  - If the (keyword, label) is not found, we set the corresponding frequency equal to 1 (10-11).
  - Otherwise, we increase the frequency by 1 (13).

```
Algorithm 1 Model training
1. initialize $\mathcal{K}$, $A$, $J$
2. for each paper $p \in T$
   $C^p \leftarrow $ ExtractResearchAreas($p$)
3. $K^p \leftarrow $ ExtractKeywords($p$)
4. for each keyword $k \in K^p$
   $|P^k| \leftarrow |P^k| + 1$
5. for each research area $c \in C^p$
   - Create pair $(k,c)$
   - if $\mathcal{K}.search(k,c) =$false
     $\mathcal{K}.insert(k,c)$
   - else
     $|P^k,c| \leftarrow |P^k,c| + 1$
```
Model Training (Phase 2)

- In this phase we populate the $A$ RDV.
- We work similarly to the previous case, but we take into account co-authorship data.
- Apart from (author, label) pairs, we also store (author, co-author, label) triples.
Vector $\mathcal{A}$

- The authors usually publish articles in more research fields.
- The produced RDV $\mathcal{A}$ stores information which demonstrates:
  - How frequently each author $a$ has been correlated with each label $c$.
  - How frequently each author $a$ has been correlated with each label $c$ when co-authored articles with $a'$. 
Co-authorship in Vector $A$

- For instance, when an arbitrary author $A$ co-operates with $B$, s/he publishes articles related to IR, whereas when co-operates with $C$, publishes articles related to DM.

- Consequently, when we classify an unlabeled article authored by $A$ and $B$, we know that probably the article discusses an IR topic.
Model Training (Phase 3)

- In this phase we populate the $\mathcal{J}$ RDV.
- There is only one journal $j$, hence, the process is simpler.
- For each pair $(j,c)$ we search within $\mathcal{J}$
  - If the (journal, label) is not found, we set the corresponding frequency equal to 1 (36-37).
  - Otherwise, we increase the frequency by 1 (39).
Classification Process

- The three relevance description vectors $K$, $A$, and $J$ are now used to label to the unclassified articles.

- For each article, each label is assigned three partial scores according to the article’s keywords, authors, journals, and the contents of $K$, $A$, and $J$.

- The final score is a linear combination of the three partial scores.
Articles Classification (Phase 1)

- In the first phase we extract the keywords of the unlabeled item.
- For each keyword $k$ we do a search in $\mathcal{K}$.
- If $k \in \mathcal{K}$, we retrieve the list of the correlated labels and for each label $c$ we compute its partial score $S_k^c$ by using:
  - The frequencies $|P^k|$, $|P^{k,c}|$.
  - A scoring function $F_k$ (i.e. IDF, $\log|P^{k,c}|/|P^k|$).

1. for each unlabeled article $p$
   2. $K^p \leftarrow $ ExtractKeywords($p$)
   3. for each keyword $k \in K^p$
   4. if $k \in \mathcal{K}$
   5. for each $(k, c) \in \mathcal{K}$
   6. $S_k^c \leftarrow F_k(P^k, P^{k,c})$
Articles Classification (Ph. 2-a)

- In this phase we use the authors vector $\mathcal{A}$.

- Initially, we check if each paper author $a$ has co-operated with the other authors.

- In case a pair $(a, a') \in \mathcal{A}$ we compute the partial score $S_a^c$ of each label $c$ by retrieving the corresponding co-authorship frequencies.
Articles Classification (Ph. 2-b)

- **Co-Authorship Frequencies**
  - $|P_{aa'}|$: The articles coauthored by $a$ and $a'$.  
  - $|P_{aa'c}|$: » » » which are labeled as $c$.

- In the opposite case where $a$ has not cooperated with any of the other authors, we compute the label score $S_a^c$ by using the plain author frequencies:
  - $|P_a|$: The number of articles authored by $a$.
  - $|P_{a,c}|$: » » » which are labeled as $c$. 
Articles Classification (Phase 3)

- Finally, we exploit the history of the article’s publishing journal.
- The score is computed by using the frequencies stored in the $J$ RDV:
  - $|P_i^j|$: The number of articles published by $j$.
  - $|P_i^j,c|$: » » » which are labeled as $c$. 

<table>
<thead>
<tr>
<th>Phase 3: Journal-based classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>19. $j \leftarrow \text{ExtractJournal}(p)$</td>
</tr>
<tr>
<td>20. if $j \in J$</td>
</tr>
</tbody>
</table>
| 21. for each $(j, c) \in J$
| 22. $S_j^c \leftarrow F_j(P_j^j, P_j^j,c)$ |
Label scores

- The label score is computed as a linear combination of the three partial scores:
  \[ S^c = w_k S_k^c + w_a S_a^c + w_j S_j^c \]

- \( w_k, w_a, w_j \) are constants used to tune the contribution of keywords, authors, journals
  \[ w_k + w_a + w_j = 1 \]
Experimental Setup

- We used CiteSeerX dataset, a collection comprised of 1.8 million research papers.
- We used three sets of labels, all based on the IEEE/ACM taxonomy:
  - C11: A set of the 11 top-level categories.
  - C81: A set of 81 mid-level categories.
  - C276: A set of 276 third-level categories.
- IEEE/ACM labeled articles: 1.1 million
Experiments

- We compared our algorithm against the state-of-the-art ML methods: SVM and AdaBoost.MH.
- We created three training sets of 10,000, 100,000 and 1.1 million articles.
- Statistics for the “large” training set:

<table>
<thead>
<tr>
<th>Vector</th>
<th>Records</th>
<th>Most Frequent</th>
<th>Articles</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K$</td>
<td>475,308</td>
<td>system</td>
<td>144,295</td>
</tr>
<tr>
<td>$A$</td>
<td>497,604</td>
<td>Philip S. Yu</td>
<td>654</td>
</tr>
<tr>
<td>$J$</td>
<td>3,915</td>
<td>Theor. Computer Science</td>
<td>13,295</td>
</tr>
</tbody>
</table>

Table 3: Trained Model Statistics
Evaluation

- We split the training set in three equally sized parts.
- We used the first two thirds to build \( K, A, \) and \( J. \)
- The last third was used for evaluation.
- We checked all the possible combinations for \( w_k, w_a, w_j. \)
Accuracy

- Our approach outperformed the adversary generic ML algorithms.
- Our method: 81%-96%
- SVM: 78%-94%.
- ADA: 80%-88%

Table 2: Optimal tuning of the $w_k$, $w_a$, and $w_j$ parameters for the three employed taxonomy structures and for training sets of different sizes.
Conclusions

- We presented a supervised machine learning approach for classifying research articles.
- Our algorithm takes into consideration the specific aspects of this particular problem.
- It outperforms the adversary approaches:
  - Better performance by about 6%, whereas AdaBoost fails to complete in large datasets.
Thank you for watching

Any questions?