28th ACM Symposium on Applied Computing
Coimbra, Portugal, 18-22 March 2013

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).

Algorithm 1 Model training

1. **initialize** $\mathcal{K}$, $\mathcal{A}$, $\mathcal{J}$
2. **for each** paper $p \in \mathcal{T}$
3. $\quad C^p \leftarrow \text{ExtractResearchAreas}(p)$

Phase 1: Processing of the keywords

4. $\quad K^p \leftarrow \text{ExtractKeywords}(p)$
5. $\quad$ **for each** keyword $k \in K^p$
6. $\quad\quad |P^k| \leftarrow |P^k| + 1$
7. $\quad\quad$ **for each** research area $c \in C^p$
8. $\quad\quad\quad$ Create pair (k, c)
9. $\quad\quad\quad$ **if** $\mathcal{K}.\text{search}(k, c) =$**false**
10. $\quad\quad\quad\quad \mathcal{K}.\text{insert}(k, c)$
11. $\quad\quad\quad\quad |P^{k,c}| \leftarrow 1$
12. $\quad\quad\quad$ **else**
13. $\quad\quad\quad\quad |P^{k,c}| \leftarrow |P^{k,c}| + 1$

- 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).

Model Training (Phase 2)

- In this phase we populate the $\mathcal{A}$ RDV.
- We work similarly to the previous case, but we take into account co-authorship data.

Phase 2: Processing of the authors

$A^p \leftarrow$ ExtractAuthors(p)
for each author $a \in A^p$
  $|P^a| \leftarrow |P^a| + 1$
  for each research area $c \in C^p$
    Create pair (a, c)
    if $\mathcal{A}.AP$.search(a, c) =**false**
      $\mathcal{A}.AP$.insert(a, c)
      $|P^{a,c}_{AP}| \leftarrow 1$
    else
      $|P^{a,c}_{AP}| \leftarrow |P^{a,c}_{AP}| + 1$
    for each author $a' \in A^p$
      Create tuple (a, a', c)
      if $\mathcal{A}.AA$.search(a, a', c) =**false**
        $\mathcal{A}.AA$.insert(a, a', c)
        $|P^{a,c}_{AA}| \leftarrow 1$
      else
        $|P^{a,c}_{AA}| \leftarrow |P^{a,c}| + 1$

- 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 $\mathcal{A}$

- 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).

Phase 3: Processing of the journals

31. $j \leftarrow$ ExtractJournal(p)
32. $|P^j| \leftarrow |P^j| + 1$
33. **for each** research area $c \in C^p$
34.   Create pair (j, c)
35.   **if** $\mathcal{J}$.search(j, c) =**false**
36. $\mathcal{J}$.insert(j, c)
37. $|P^{j,c}| \leftarrow 1$
38.   **else**
39. $|P^{j,c}| \leftarrow |P^{j,c}| + 1$

Classification Process

- The three relevance description vectors $\mathcal{K}$, $\mathcal{A}$, and $\mathcal{J}$ 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 $\mathcal{K}$, $\mathcal{A}$, and $\mathcal{J}$.
- The final score is a linear combination of the three partial scores.

Articles Classification (Phase 1)

for each unlabeled article p

Phase 1: Keyword-based classification

$K^p \leftarrow \text{ExtractKeywords}(p)$
  for each keyword $k \in K^p$
    if $k \in \mathcal{K}$
      for each $(k, c) \in \mathcal{K}$
        $\mathcal{S}_k^c \leftarrow F_k(P^k, P^{k,c})$

- 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 $\mathcal{S}_k^c$ by using:
  - The frequencies $|P^k|$, $|P^{k,c}|$.
  - A scoring function F_k (i.e. IDF, $\log|P^{k,c}|/|P^k|$).

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.

Phase 2: Author-based classification

```
A^p ← ExtractAuthors(p)
for each author a ∈ A^p
    coauthor ← false
    for each author a' ∈ A^p
        if (a, a') ∈ 𝒜.AA
            coauthor ← true
            for each (a, a', c) ∈ 𝒜.AA
                S_a^c ← F_a(P^a_{AA}, P^{a,c}_{AA})
    if coauthor = false
        if a ∈ 𝒜.AP
            for each (a, c) ∈ 𝒜.AP
                S_a^c ← F_a(P^a_{AP}, P^{a,c}_{AP})
```

- In case a pair *(a,a')* $\in \mathcal{A}$ we compute the partial score $\mathcal{S}_a^c$ 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 co-operated with any of the other authors, we compute the label score $\mathcal{S}_a^c$ 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.

Phase 3: Journal-based classification

$j \leftarrow \text{ExtractJournal}(p)$
if $j \in \mathcal{J}$
  for each $(j, c) \in \mathcal{J}$
    $\mathcal{S}_j^c \leftarrow F_j(P^j, P^{j,c})$

- The score is computed by using the frequencies stored in the $\mathcal{J}$ RDV:
  - $|P^j|$: The number of articles published by *j*.
  - $|P^{j,c}|$: » » » which are labeled as *c*.

Label scores

- The label score is computed as a linear combination of the three partial scores:

$$\mathcal{S}^c = w_k \mathcal{S}_k^c + w_a \mathcal{S}_a^c + w_j \mathcal{S}_j^c$$

- 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:

Vector	Records	Most Frequent	Articles
$\mathcal{K}$	475,308	system	144,295
$\mathcal{A}$	497,604	Philip S. Yu	654
$\mathcal{J}$	3,915	Theor. Computer Science	13,295

Table 3: Trained Model Statistics

Evaluation

- We split the training set in three equally sized parts.
- We used the first two thirds to build $\mathcal{K}$, $\mathcal{A}$, and $\mathcal{J}$.
- 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%
  - Some experiments did not finish!

$\|T\|$	C	$\{w_k, w_a, w_j\}$	**Acc.**	**SVM**	**Ada**
10,000	$C11$	{0.3, 0.1, 0.6}	94.0%	88.2%	88.8%
	$C81$	{0.2, 0.1, 0.7}	87.5%	82.9%	83.4%
	$C276$	{0.2, 0.1, 0.7}	80.7%	78.4%	80.1%
100,000	$C11$	{0.3, 0.2, 0.5}	95.1%	89.6%	-
	$C81$	{0.3, 0.1, 0.6}	88.2%	84.3%	-
	$C276$	{0.2, 0.2, 0.6}	80.9%	79.0%	-
1,159,634	$C11$	{0.3, 0.2, 0.5}	95.9%	94.1%	-
	$C81$	{0.3, 0.2, 0.5}	89.0%	87.9%	-
	$C276$	{0.3, 0.1, 0.6}	81.3%	80.8%	-

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?