A Tensor-based Clustering Approach for Multiple Document Classifications

Salvatore Romeo[1], Andrea Tagarelli[1], Francesco Gullo[2] and Sergio Greco[1]
[1]*DIMES Dept., University of Calabria, Cosenza, Italy*
[2]*Yahoo! Research, Barcelona, Spain*

Keywords: Document Clustering, Itemset Mining, Tensor Modeling and Decomposition.

Abstract: We propose a novel approach to the problem of document clustering when multiple organizations are provided for the documents in input. Besides considering the information on the text-based content of the documents, our approach exploits frequent associations of the documents in the groups across the existing classifications, in order to capture how documents tend to be grouped together orthogonally to different views. A third-order tensor for the document collection is built over both the space of terms and the space of the discovered frequent document-associations, and then it is decomposed to finally establish a unique encompassing clustering of documents. Preliminary experiments conducted on a document clustering benchmark have shown the potential of the approach to capture the multi-view structure of existing organizations for a given collection of documents.

1 INTRODUCTION

Real-world data often presents inherent characteristics that raise challenging issues to their effective analysis, namely high dimensionality and multi-faceted nature. *Tensor representation and decompositions* (Cichocki et al., 2009; Kolda and Bader, 2009) are natural approaches for handling large amounts of such data. They are indeed considered as a multilinear generalization of matrix factorizations, since all dimensions or modes are retained thanks to multilinear models which can produce unique and meaningful components.

There exists a large variety of application domains in which tensor representation and decompositions are being increasingly applied, ranging from chemometrics and psychometrics to signal processing, from computer vision to neuroscience and image recognition (Kolda and Bader, 2009). The applicability of tensor models has also attracted growing attention in pattern recognition, information retrieval, and data mining related fields to solve problems such as dimensionality reduction (Liu et al., 2005), link analysis (Kolda and Bader, 2006), and document clustering (Liu et al., 2011; Kutty et al., 2011). Focusing on the clustering task, advanced methods can go beyond the usual approach that yields a partition of the input dataset to overlapping, fuzzy, or probabilistic clustering; however, many real-world clustering-based applications are increasingly demanding for taking into account some knowledge about the multi-faceted nature of a document collection, for which multiple predefined organizations might be available.

In this paper we are interested in extending the task of document clustering, which is traditionally performed according only to the textual content information of the documents, to the case in which a single clustering is desired starting from multiple organizations of the documents. Such existing document organizations can be seen as multiple views over a document collection which might correspond to user-provided, possibly alternative organizations, or to the results separately obtained by one or more document clustering algorithms or supervised text classifiers. For example, news articles can be clustered based on the topics they discuss, or to reflect some existing categorization of major themes or meta-information they are related to.

The underlying assumption of our approach is that, when the documents can be naturally grouped in multiple ways, a single new clustering encompassing all existing document organizations can be obtained by integrating the textual content information with knowledge on the available groupings of the documents. However, since no information about class labels of the available groupings is assumed to be avail-

Romeo S., Tagarelli A., Gullo F. and Greco S. (2013).
A Tensor-based Clustering Approach for Multiple Document Classifications.
In *Proceedings of the 2nd International Conference on Pattern Recognition Applications and Methods*, pages 200-205
DOI: 10.5220/0004269102000205

able, our key idea to accomplish the task relies on the identification of frequent co-occurrences of documents in the groups across the existing organizations, in order to capture how documents tend to be grouped together orthogonally to the different views. Based on the discovered frequent associations of the documents as well as on the usual term-document representation of the text contents, a novel tensor model is built and decomposed to finally establish a unique clustering of documents that might be suited to reflect the multi-dimensional structure of the initial document organizations.

To the best of our knowledge, no other existing tensor-based approach for document clustering is conceived to handle the availability of multiple organizations for the documents in input. We would like also to point out that, while the problem of extracting a single clustering from multiple existing ones is actually not novel—a large corpus of research in advanced data clustering has been developed to address the problem of *ensemble clustering* (see (Ghosh and Acharya, 2011) for an overview)—in this work we face the problem from a different perspective, which relies on a tensorial representation of a set of clusterings and also relaxes a main assumption in ensemble clustering methods, whereby the feature relevance values are assumed to be unavailable.

2 PROPOSED APPROACH

We are given a collection $\mathcal{D} = \{D_1, \ldots, D_{|\mathcal{D}|}\}$ of documents, which are represented over a set $\mathcal{V} = \{w_1, \ldots, w_{|\mathcal{V}|}\}$ of terms. We are also given a set of organizations of the documents in $\mathcal{D}$, denoted as $\mathcal{CS} = \{\mathcal{C}_1, \ldots, \mathcal{C}_H\}$, such that each $\mathcal{C} \in \mathcal{CS}$ represents a set of homogeneous groups of documents. We hereinafter generically refer to each of the document organizations as a *document clustering* and to each of the homogeneous groups of documents as *document cluster*.

In the following we describe in detail the proposed framework, broken down into four main steps, namely extraction of closed document-sets from multiple document organizations, construction of the tensor model, decomposition of the tensor, and induction of a document clustering.

2.1 Extracting Closed Frequent Document-sets

In our setting, an item corresponds to a document in $\mathcal{D}$, hence an itemset is a *document-set*, while a transaction corresponds to a cluster that belongs to any clustering in $\mathcal{CS}$. As a transactional dataset is a multi-set of transactions, there will be as many transactions as the number of clusters over all document clusterings in $\mathcal{CS}$.

We extract document-sets that frequently occur (given a user-specified minimum-support threshold) over all available clusters, specifically frequent document-sets that are *closed*, since we aim to minimize the size of the set of patterns discovered while ensuring its completeness. However, in contrast to typical scenarios of transactional data, a peculiarity of our context is that the size of the transactional dataset (i.e., the number of document clusters) is much lower than the size of the item domain (i.e., the number of documents). Thus, in order to extract (closed) frequent document-sets, a traditional (closed) frequent itemset mining approach could be prohibitive, as it would require a cost which is exponential with the number of documents.

Given a transactional dataset $\mathcal{T}$, we denote with t a transaction, with $T \subseteq \mathcal{T}$ a set of transactions (transaction-set), and with $I_T = \bigcap_{t \in T} t$ the itemset shared by the transactions (i.e., clusters) in T. A transaction-set containing d transactions is said a d-transaction-set, with $1 \leq d \leq |\mathcal{T}|$. Figure 1 shows the proposed closed frequent itemset miner, which uses a level-wise search where d-transaction-sets are used to explore $(d+1)$-transaction-sets. To perform the search, an enumeration tree is incrementally built such that each node represents a pair of the form (T, I_T). The initial set of 1-transaction-sets (Line 2) is used to compute $(1+it)$-transaction-sets, at each iteration *it* of the *search* procedure; this procedure terminates after at most $|\mathcal{T}|$ levels along each branch of the enumeration tree. To avoid redundant unions among transaction-sets (hence, intersections among their itemsets), the ordering between the first transactions of any two transaction-sets is involved at each iteration (Lines 8 and 10). Note that, as the search space is being explored, the support of the itemsets obtained by the intersection of a growing number of transactions is monotonically non-decreasing. Therefore, every candidate closed itemset (Line 11) is checked to be a frequent itemset (Line 13). The *merge* function (Line 5) searches for all pairs that have the same common itemset and yields a single pair containing the union of the transaction-sets. Finally, the set *CI* of all closed frequent itemsets is created from the set of I_T elements in $C_{IT\text{-}P}$ (function *extractIT*, Line 6).

Input:
A transactional dataset $\mathcal{T}$,
a minimum support threshold *minsup*
Output:
A set *CI* of closed frequent itemsets
begin
1. $C_{IT\text{-}P} \leftarrow \emptyset$
2. $P \leftarrow \{(\{t\},t) \mid t \in \mathcal{T}\}$
3. $P_0 \leftarrow P$
4. *search*$(P_0,P,C_{IT\text{-}P})$
5. $C_{IT\text{-}P} \leftarrow merge(C_{IT\text{-}P})$
6. $CI \leftarrow extractIT(C_{IT\text{-}P})$

end

procedure *search*$(P',P,C_{IT\text{-}P})$
7. **for all** $(T,I_T) \in P'$ **do**
8. let t be the first transaction in T
9. $P'' \leftarrow \emptyset$
10. **for all** $(\{t_i\},t_i) \in P, t < t_i$ **do**
11. $T_j \leftarrow T \cup \{t_i\}$, $I_{T_j} \leftarrow I \cap \{t_i\}$
12. $P'' \leftarrow P'' \cup \{(T_j,I_{T_j})\}$
13. **if** $support(I_{T_j}) \geq minsup$ **then**
14. remove from $C_{IT\text{-}P}$ all (T_k,I_{T_k}) such that
15. $T_j \supseteq T_k$ and $I_{T_k} = I_{T_j}$
16. **if** I_{T_j} is a closed itemset for the itemsets
17. in $C_{IT\text{-}P}$ **then**
18. $C_{IT\text{-}P} \leftarrow C_{IT\text{-}P} \cup \{(T_j,I_{T_j})\}$
19. **endIf**
20. **endIf**
21. **endFor**
22. *search*$(P'',P,C_{IT\text{-}P})$
23. **endFor**

Figure 1: Intersection-based closed frequent itemset mining algorithm.

2.2 Building a Tensor for Multiple Document Organizations

To model a set of documents contextually to multiple available organizations of the documents, we define a *third-order tensor* such that the first mode corresponds to the closed frequent document-sets extracted from the set of document organizations, the second mode to the terms representing the document contents, and the third mode to the documents.

Formally, if we denote with $\mathcal{CDS} = \{CDS_1,\ldots,CDS_{|\mathcal{CDS}|}\}$ the set of closed frequent document-sets extracted from $\mathcal{CS}$, we define a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, where $I_1 = |\mathcal{CDS}|$, $I_2 = |\mathcal{V}|$, and $I_3 = |\mathcal{D}|$. Hence, the i_3-th slice of the tensor refers to document D_{i_3} and is represented by a matrix of size $I_1 \times I_2$, where the (i_1,i_2)-th entry will be computed to determine the relevance of term w_{i_2} in document D_{i_3} contextually to the document-set CDS_{i_1}.

Given a document D, a term w, and a frequent document-set *CDS* (we omit here the subscripts for the sake of readability of the following formulas), our aim is to incorporate the following aspects in the term relevance weight: (1) the popularity of the term in the document, (2) the rarity of the term over the collection of documents, (3) the rarity of the term locally to the frequent document-set, and (4) the support of the frequent document-set.

Aspects 1 and 2 refer to the notions of *term frequency* and *inverse document frequency* in the classic $tf.idf$ term relevance weighting function. Formally, the frequency of term w in document D, denoted as $tf(w,D)$, is equal to the number of occurrences of w in D. The inverse document frequency of term w in the document collection is defined as $idf(w) = \log(|\mathcal{D}|/N(w))$, where $N(w)$ is the number of documents in $\mathcal{D}$ that contain w.

To account for aspect 3, we introduce an *inverse document-set frequency* factor: $idsf(w,CDS) = \log(1 + |CDS|/N(w,CDS))$, where $|CDS|$ is the number of documents belonging to the frequent document-set *CDS*, and $N(w,CDS)$ denotes the number of documents in *CDS* that contain w. Moreover, the $idsf$ weight is defined to be equal to zero if term w is absent in all documents of *CDS*; otherwise, note that $idsf$ weight is always a positive value even in case of maximum popularity of the term in the frequent document-set.

As for aspect 4, we exploit the support of the frequent document-set: $s(CDS) = \exp(supp(CDS)/(\max_{CDS' \in \mathcal{CDS}} supp(CDS')))$, where $supp(CDS)$ is the support of *CDS*, i.e., the number of clusters in every $\mathcal{C} \in \mathcal{CS}$ that contain *CDS*. Note that the support of a document-set is bounded by $|\mathcal{CS}|$ in case of non-overlapping clusters in each $\mathcal{C}$.

Finally, by combining all four factors, the overall term relevance weighting function has the form: $weight(CDS,w,D) = tf(w,D)\ idf(w)\ idsf(w,CDS)\ s(CDS)$.

2.3 Tensor Decomposition

According to the tensor decomposition notations in (Cichocki et al., 2009), we will use symbols $\otimes$, $\circledast$, and $\times_n$ to denote the Kronecker product, the element-wise product, and the mode-n product (with $n \in \{1,2,3\}$), respectively. Moreover, if we denote with $\mathcal{G}$ the core tensor and with $\mathbf{A}$ its factor matrices, symbols $\hat{\mathcal{X}} = \mathcal{G} \times \{\mathbf{A}\}$, $\mathbf{A}^{\otimes -n}$, $\mathcal{G} \times_{-n} \{\mathbf{A}\}$ and $\mathbf{X}_{(n)}$ denote the product $\mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_3 \mathbf{A}^{(3)}$, the Kronecker product between all factor matrices except $\mathbf{A}^{(n)}$, the mode-n product between $\mathcal{G}$ and all factor matrices except $\mathbf{A}^{(n)}$ and the matricization along mode n of tensor $\mathcal{X}$, respectively.

Nonnegative Tucker Decomposition (NTD) is the state-of-the-art in tensor decomposition, which allows for taking into account all interactions between the tensor modes. Particularly, we refer to the *fast Be-*

Input:
$\mathcal{X}$: input data of size $I_1 \times I_2 \times I_3$,
J_1, J_2, J_3: number of basis for each factor,
β: divergence parameter (default: 2).
Output:
core tensor $\mathcal{G} \in \mathbb{R}^{J_1 \times J_2 \times J_3}$
factor matrices $\mathbf{A}^{(1)} \in \mathbb{R}_+^{I_1 \times J_1}$, $\mathbf{A}^{(2)} \in \mathbb{R}_+^{I_2 \times J_2}$, $\mathbf{A}^{(3)} \in \mathbb{R}_+^{I_3 \times J_3}$
begin
1. Nonnegative ALS initialization for all $\mathbf{A}^{(n)}$ and $\mathcal{G}$
2. **repeat**
3. $\hat{\mathcal{X}}' = \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)}$
4. $\mathcal{X}''$ = *computeStep1*($\mathcal{X}$,$\hat{\mathcal{X}}'$,$\mathbf{A}^{(3)}$,n,β)
5. **for** n = 1 to 3 **do**
6. $\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)} \circledast$ *computeStep2*($\mathcal{X}''$,$\mathbf{A}$,$\mathcal{G}$,n) $\oslash$ *computeStep3*($\hat{\mathcal{X}}'$,$\mathbf{A}$,$\mathcal{G}$,β,n)
7. $\mathbf{a}_{j_n}^{(n)} \leftarrow \mathbf{a}_{j_n}^{(n)} / \| \mathbf{a}_{j_n}^{(n)} \|_p$
8. **endFor**
9. $\mathcal{G} \leftarrow \mathcal{G} \circledast \left[\mathcal{X}'' \times \{\mathbf{A}^T\}\right] \oslash \left[\hat{\mathcal{X}}^{\cdot[\beta]} \times \{\mathbf{A}^T\}\right]$
10. **until** *a stopping criterion is met*
end

Figure 2: Modified fast BetaNTD algorithm.

taNTD algorithm (Cichocki et al., 2009) that relies on beta divergences (Basu et al., 1998), which have been successfully applied for robust PCA and clustering. The fast BetaNTD algorithm has multiplicative update rules defined in function of the tensor $\mathcal{X}$ and its current approximation $\hat{\mathcal{X}}$. Unfortunately, $\hat{\mathcal{X}}$ is a large yet dense tensor and hence it cannot be easily kept in primary memory. We decompose the tensor following the lead of the approach proposed in (Kolda and Sun, 2008), and our resulting algorithm is shown in Figure 2. To avoid storing the entire tensor $\hat{\mathcal{X}}$, we keep in memory only an intermediate result $\hat{\mathcal{X}}' = \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)}$ (Line 3), and then partially compute the final approximated tensor as $\hat{\mathcal{X}} = \hat{\mathcal{X}}' \times_3 \mathbf{A}^{(3)}$ only for a limited number of entries at time, for each mode.

Let us consider the update rule for any $\mathbf{A}^{(n)}$:

$$\mathbf{A}^{(n)} \leftarrow \mathbf{A}^{(n)} \circledast \left[\left(\mathbf{X}_{(n)} \circledast \hat{\mathbf{X}}_{(n)}^{\cdot[\beta-1]}\right)\mathbf{A}^{\otimes -n}\mathbf{G}_{(n)}^T\right] \oslash \left(\hat{\mathbf{X}}_{(n)}^{\cdot[\beta]}\mathbf{A}^{\otimes -n}\mathbf{G}_{(n)}^T\right)$$

In the above rule, the most expensive operations are $\mathbf{X}_{(n)} \circledast \hat{\mathbf{X}}_{(n)}^{\cdot[\beta-1]}$, $\mathbf{A}^{\otimes -n}\mathbf{G}_{(n)}^T$, and $\hat{\mathbf{X}}_{(n)}^{\cdot[\beta]}\mathbf{A}^{\otimes -n}\mathbf{G}_{(n)}^T$, thus we decompose the problem into three smaller steps: (1) computation of $\mathcal{X} \circledast \hat{\mathcal{X}}^{\cdot[\beta-1]}$ (which considers only the nonzero entries of $\mathcal{X}$), (2) block-wise computation of $(\mathbf{X}_{(n)} \circledast \hat{\mathbf{X}}_{(n)}^{\cdot[\beta-1]})\mathbf{A}^{\otimes -n}\mathbf{G}_{(n)}^T$, and (3) block-wise computation of $\hat{\mathbf{X}}_{(n)}^{\cdot[\beta]}\mathbf{A}^{\otimes -n}\mathbf{G}_{(n)}^T$.

Step 1. Product $\mathcal{X} \circledast \hat{\mathcal{X}}^{\cdot[\beta-1]}$ is computed once to obtain matricizations given by $\mathbf{X}_{(n)} \circledast \hat{\mathbf{X}}_{(n)}^{\cdot[\beta-1]}$. Moreover, the product $\mathcal{X} \circledast \hat{\mathcal{X}}^{\cdot[\beta-1]}$ is computed starting from the intermediate result $\hat{\mathcal{X}}'$ and, since $\mathcal{X} \circledast \hat{\mathcal{X}}^{\cdot[\beta-1]}$ is the element-wise product of a sparse tensor with a dense one, the resulting tensor will also be a sparse tensor whose nonzero entries are in the same positions as those within $\mathcal{X}$, and only these entries need to be computed.

Step 2. Since $\mathbf{A}^{\otimes -n}\mathbf{G}_{(n)}^T$ is the transpose of the matricization along the mode n of the tensor resulting from $\mathcal{G} \times_{-n} \{\mathbf{A}\}$, which has the same number of columns of $\mathbf{X}_{(n)} \circledast \hat{\mathbf{X}}_{(n)}^{\cdot[\beta-1]}$, it can be noted that $(\mathbf{X}_{(n)} \circledast \hat{\mathbf{X}}_{(n)}^{\cdot[\beta-1]})\mathbf{A}^{\otimes -n}\mathbf{G}_{(n)}^T$ is the sum of a certain number of matrix products; for instance, for $n = 1$, $(\mathbf{X}_{(n)} \circledast \hat{\mathbf{X}}_{(n)}^{\cdot[\beta-1]})\mathbf{A}^{\otimes -n}\mathbf{G}_{(n)}^T$ will be the result of the sum of I_3 matrix products.

Step 3. $\hat{\mathbf{X}}_{(n)}^{\cdot[\beta]}\mathbf{A}^{\otimes -n}\mathbf{G}_{(n)}^T$ (Line 6) is computed analogously to $(\mathbf{X}_{(n)} \circledast \hat{\mathbf{X}}_{(n)}^{\cdot[\beta-1]})\mathbf{A}^{\otimes -n}\mathbf{G}_{(n)}^T$. Finally, for the core tensor update rule (Line 9), we compute a normal mode-n product and each entry of $\hat{\mathcal{X}}^{\cdot\beta}$ is computed starting from the intermediate result $\hat{\mathcal{X}}'$.

2.4 Document Clustering Induction

We consider different ways of inducing a document clustering solution from the decomposed tensor. One simple way is to derive a monothetic clustering from the third factor matrix ($\mathbf{A}^{(3)}$) by assigning each document to the component (cluster) corresponding to the highest relevance value stored in the matrix. A direct way is to input a standard document clustering algorithm with $\mathbf{A}^{(3)}$. An alternative way, which does not explicitly involve $\mathbf{A}^{(3)}$, is to consider a clustering solution obtained by applying a document clustering algorithm to the projection of the matrix of the term-frequencies (over the original document collection) to $\mathbf{A}^{(2)}$—the rationale here is to project the original document vectors of term-frequencies along the mode-2 components, which express discriminative information for the term grouping, hence deriving a clustering of the documents that are mapped to a lower dimensional space. We hereinafter refer to the different ways as *monothetic*, *direct*, and *tf-projected* document clustering, respectively.

3 EVALUATION AND RESULTS

Reuters Corpus Volume 1 (RCV1) (Lewis et al., 2004) is a major benchmark for text classification/clustering research. RCV1 is particularly suited for our study since every news, besides its plain-text fields (i.e., body and headlines) is originally provided with al-

Table 1: Document classification sets.

	news fields	*text proc. params*	*clustering params*	*size (no. of clusters)*
CS1	headline	$lf = 0$	$k \in [5..20]$	4 (50)
	body	$lf = \{0,1,5\}$	$k \in [5..20]$	12 (150)
CS2	headline + body	$lf = 5$	$k \in [5..43]$	20 (480)
CS3	headline	$lf = 0$	$k \in [5..20]$	4 (50)
	body	$lf = 0$	$k \in [5..20]$	4 (50)
	metadata	–	–	3 (19)

ternative categorizations according to three different category fields (metadata): TOPICS (i.e., major subjects), INDUSTRIES (i.e., types of businesses discussed), and REGIONS (i.e., geographic locations and economic/political side information). After filtering out very short news (i.e., documents with size less than 6KB) and any news that did not have at least one value for each of the three category fields, we selected the news labeled with one of the Top-5 categories for each of the three category fields. This resulted in a dataset of 3081 news. From the text of the news, we discarded strings of digits, retained alphanumerical terms, performed removal of stop-words and word stemming.

We generated various sets of classifications obtained over the RCV1 dataset, according to the textual content information as well as to the Topics/Industries/Regions metadata. For the purpose of generating the text-based classifications, we used the *bisecting k-means* algorithm implemented in the well-known CLUTO toolkit (Karypis, 2007) to produce clustering solutions of the documents represented over the space of the terms contained in the body and/or headlines. Table 1 summarizes the main characteristics of the three sets of document classifications used in our evaluation. Columns *text proc. params* and *clustering params* refer to the lower document-frequency cut threshold (lf, percent) used to select the terms for the document representation, and to the number of clusters (k, with increment of 5 in CS1,CS3 and 2 in CS2) taken as input to CLUTO to generate the text-based classifications. Moreover, column *size* reports the number of classifications and relating number of clusters of documents (within brackets) that rely on the same type of information (i.e., body, headline, metadata).

For each of the three document classification sets, we derived different tensors according to various settings of the closed frequent document-set extraction. Table 2 contains details about the tensors built upon the selected configurations. Note that, in each of the tensors, mode-2 corresponded to the space of terms extracted from the body and headline of the news (2692 terms) and mode-3 to the average number of clusters in the corresponding classification sets (i.e., 13 for CS1, 24 for CS2, and 11 for CS3).

Table 2: Tensors and their decompositions.

	min length of CDS	*no. of CDS*	*avg % of CDS per doc.*	_TD-S *size*
CS1_Ten1	50	17443	3.29%	$174 \times 27 \times 13$
CS1_Ten2	100	5871	5.25%	$58 \times 27 \times 13$
CS1_Ten3	150	2454	7.12%	$24 \times 27 \times 13$
CS1_Ten4	200	1265	8.53%	$12 \times 27 \times 13$
CS2_Ten1	50	12964	3.78%	$129 \times 27 \times 24$
CS2_Ten2	100	7137	4.87%	$71 \times 27 \times 24$
CS2_Ten3	150	3129	5.89%	$31 \times 27 \times 24$
CS2_Ten4	180	918	7.53%	$9 \times 27 \times 24$
CS3_Ten1	50	2806	3.09%	$28 \times 27 \times 11$
CS3_Ten2	100	843	5.15%	$8 \times 27 \times 11$
CS3_Ten3	150	326	7.15%	$3 \times 27 \times 11$

Table 3: Summary of results.

_TD-S	*clustering*	*F*	*Q*	_TD-L	*clustering*	*F*	*Q*
CS1_Ten1	monoth.	0.509	0.603	CS1_Ten1	direct	0.556	0.601
	tf-proj.	0.610	0.838		tf-proj.	0.665	0.881
CS1_Ten2	monoth.	0.534	0.599	CS1_Ten2	direct	0.570	0.603
	tf-proj.	0.625	0.838		tf-proj.	0.688	0.884
CS1_Ten3	monoth.	0.542	0.598	CS1_Ten3	direct	0.586	0.601
	tf-proj.	0.624	0.835		tf-proj.	0.689	0.889
CS1_Ten4	monoth.	0.533	0.598	CS1_Ten4	direct	0.579	0.605
	tf-proj.	0.624	0.838		tf-proj.	0.687	0.837
CS2_Ten1	monoth.	0.494	0.603	CS2_Ten1	direct	0.599	0.604
	tf-proj.	0.569	0.847		tf-proj.	0.625	0.893
CS2_Ten2	monoth.	0.496	0.603	CS2_Ten2	direct	0.556	0.601
	tf-proj.	0.561	0.843		tf-proj.	0.629	0.889
CS2_Ten3	monoth.	0.495	0.603	CS2_Ten3	direct	0.560	0.604
	tf-proj.	0.570	0.846		tf-proj.	0.635	0.895
CS2_Ten4	monoth.	0.497	0.604	CS2_Ten4	direct	0.555	0.602
	tf-proj.	0.577	0.848		tf-proj.	0.639	0.890
CS3_Ten1	monoth.	0.556	0.597	CS3_Ten1	direct	0.619	0.600
	tf-proj.	0.617	0.837		tf-proj.	0.677	0.888
CS3_Ten2	monoth.	0.556	0.597	CS3_Ten2	direct	0.619	0.599
	tf-proj.	0.620	0.837		tf-proj.	0.686	0.839
CS3_Ten3	monoth.	0.553	0.597	CS3_Ten3	direct	0.610	0.596
	tf-proj.	0.620	0.837		tf-proj.	0.680	0.887

For each of the tensors constructed, we run the algorithm in Figure 2 with different settings to obtain two decompositions: the first one led to a core-tensor with a number of components on mode-3 equal to the average number of clusters in the original classification set, whereas the other two modes were set equal to the number of closed document-sets and number of terms, respectively, scaled by a factor of 0.01; the second decomposition was devised to obtain a larger core-tensor with components of each mode equal to an increment of a multiplicative factor of 10 w.r.t. the mode in the core-tensor obtained by the first decomposition. We use suffixes _TD-S and _TD-L to denote the first (smaller) and second (larger) decompositions of a tensor, respectively; last column in Table 2 contains details about _TD-S decompositions. From the result of a _TD-S (resp. _TD-L) decomposition, we derived a monothetic (resp. direct) or, alternatively, a tf-projected clustering solution, with number of clusters equal to the number of mode-3 components.

All clustering solutions were evaluated in terms of average *F-measure* (Steinbach et al., 2000) (F) between a clustering solution derived from the tensor model and each of the input document classifications. By using the original $tf.idf$ representation of the documents (based on the text of body plus

headline fields), we also computed the centroid-based intra-cluster similarity and inter-cluster similarity and then used their difference to obtain an overall quality score (Q).

Table 3 shows our main experimental results. Comparing the performance of the different types of induced clustering, the tf-projected solutions achieved higher quality than the monothetic clusterings (for the case TD-S) and the direct clusterings (for the case TD-L), which was particularly evident in terms of internal quality. Looking at each classification-set tensors, we observed that a lower average percentage of closed document-sets generally led to slightly better performance for classification-sets characterized by conceptually different views (i.e., CS1 and CS3), whereas an inverse tendency occurred for a more homogeneous classification-set (CS2). However, we also observed no significant differences in the overall average performance obtained by varying the number of components in mode-1, which would indicate a relatively small sensitivity of the tensor approximation to the mode-1 (i.e., space of the mined closed document-sets). Also, the F-measure scores for the CS3 tensors were comparable or even better than for the other tensors, which would suggest the ability of our tensor model to handle document classification sets which express possibly alternative views (i.e., different content-based views along with metadata-based views).

4 CONCLUSIONS

We proposed a novel document clustering framework that copes with the knowledge about multiple existing classifications of the documents. The main novelty of our study is the definition of a third-order tensor model that takes into account both the document content information and the ways the documents are grouped together across the available classifications. Further work might be focused on a thorough investigation of the impact of the frequent document-sets on the sparsity of the tensor, and on a comparison with clustering ensemble methods.

REFERENCES

Basu, A., Harris, I. R., Hjort, N. L., and Jones, M. C. (1998). Robust and efficient estimation by minimising a density power divergence. *Biometrika*, 85(3):pp. 549-559.

Cichocki, A., Phan, A. H., and Zdunek, R. (2009). *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. Wiley, Chichester.

Ghosh, J. and Acharya, A. (2011). Cluster ensembles. *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery*, 1(4):305-315.

Karypis, G. (2002/2007). CLUTO - Software for Clustering High-Dimensional Datasets. http://www.cs.umn.edu/ cluto.

Kolda, T. and Bader, B. (2006). The TOPHITS model for higher-order web link analysis. In *Proc. SIAM Workshop on Link Analysis, Counterterrorism and Security*.

Kolda, T. G. and Bader, B. W. (2009). Tensor decompositions and applications. *SIAM Review*, 51(3):455-500.

Kolda, T. G. and Sun, J. (2008). Scalable tensor decompositions for multi-aspect data mining. In *Proc. IEEE ICDM Conf.*, pages 363-372.

Kutty, S., Nayak, R., and Li, Y. (2011). XML Documents Clustering Using a Tensor Space Model. In *Proc. PAKDD Conf.*, pages 488-499.

Lewis, D. D., Yang, Y., Rose, T., and Li, F. (2004). RCV1: A New Benchmark Collection for Text Categorization Research. *Journal of Machine Learning Research*, 5:361-397.

Liu, N., Zhang, B., Yan, J., Chen, Z., Liu, W., Bai, F., and Chien, L. (2005). Text Representation: From Vector to Tensor. In *Proc. IEEE ICDM Conf.*, pages 725-728.

Liu, X., Glänzel, W., and Moor, B. D. (2011). Hybrid clustering of multi-view data via Tucker-2 model and its application. *Scientometrics*, 88(3):819-839.

Steinbach, M., Karypis, G., and Kumar, V. (2000). A Comparison of Document Clustering Techniques. In *Proc. KDD Text Mining Workshop*.