FAST SUPERVISED FEATURE EXTRACTION FROM STRUCTURED REPRESENTATION OF TEXT DATA

Ondřej Háva[1], Mirek Skrbek[2], Pavel Kordík[3]

[1]Department of Computer Science and Engineering, Faculty of Electrical Engineering, Czech Technical University in Prague
Zikova 1905/4, 166 36 Prague 6, Czech Republic
[2,3]Department of Computer Science, Faculty of Information Technology, Czech Technical University in Prague
Zikova 1905/4, 166 36 Prague 6, Czech Republic

havaondr@fel.cvut.cz

Abstract

Classification of text documents is challenging problem not only when browsing the web. Structure representation of documents is necessary to build up appropriate classifier. Unfortunately the document-term matrices are usually so sparse and of high dimensionality due large number of terms representing usually smaller number of the documents. Thus the suitable dimensionality reduction technique is required to be able to develop the classifier. The article deals with supervised extraction method that results to small number of sensitive features derived from the initial document-term matrix. The extraction process simulated by neural network is remarkably fast and utilizes all available supervised information from training data.

Keywords: Dimensionality reduction, Document representation, Fast network learning.

Presenting Author's biography

Ondřej Háva. Part-time postgraduate student. He primarily focuses on text mining and machine learning.

1 Introduction

Unstructured text data must be transformed to structure representation to enable predictive models to classify the documents. Written language is very rich and there are a lot of words and phrases in the text. Thus a lot of features describing the documents can be derived. Developing the models over such high dimensional data is usually impossible therefore the algorithms for dimensionality reduction must be inserted into the process before predictive modeling. When the objective of the modeling is known one can take advantage of the identified target to derive or select small number of features with optimized predictive power.

It is not appropriate to perform dimensionality reduction in one step especially when the reduction is significant. Thus we tested two-stage algorithm that leverages from the similarities with training categorized documents in the first stage. The categorization of training documents is utilized in the second stage.

2 Document representation

Let us have a collection of text documents. Each document can be parsed into set of strings that describes the document [8]. Strings can be formed by words occurring in documents or by consecutive sequences of characters called n-grams. Words can be transformed to their basic forms by linguistic algorithms or multiword concepts can be derived from them. Later in text we will refer to extracted features as terms. Terms can be words, n-grams or concepts.

The terms used for description of document collection form the dictionary [2]. Terms extracted from collection are usually reduced by frequency filter and stop word list is applied before they are added to the dictionary [7, 8]. Dictionary items form the columns of document-term matrix. Document-term matrix serves as structured representation of unstructured text documents.

Documents are located in row vectors in document-term matrix. Items of document vector are called weights [7]. In the simplest case the weights are binary indicators of the presence or absence of a particular term in a document. More commonly the weight represents the frequency of the term in the document. Transformation of weights can be employed to scale the term frequencies regarding to uniqueness of the term in the document. Normalization of weights can be applied as well to adjust for different lengths of documents.

		terms (words, n-grams, concepts, ...)				
		t_1	t_2	t_3	…	t_M
documents	d_1	w_{11}	w_{12}	w_{13}	…	w_{1M}
	d_2	w_{21}	w_{22}	w_{23}	…	w_{2M}
	⋮	⋮	⋮	⋮	…	⋮
	d_N	w_{N1}	w_{N2}	w_{N3}	…	w_{NM}

Fig. 1 document-term matrix

3 Dimensionality reduction techniques

Due the written language richness document collections are described by large number of features. Usually the number of terms extracted from the collection is larger than number of the documents. Developing the model over such data matrix is problematic and time consuming or even it can be impossible due model assumptions. The relationships among the extracted terms can also degrade the performance of the model. Thus the dimensionality reduction techniques are frequently applied before the model is developed.

There are two main approaches to dimensionality reduction [1]. *Feature selection* techniques pick several important features out of large set of former features. *Feature extraction* methods derive new set of features from the original set. New extracted features represent the original features as accurate as possible but in a smaller number of dimensions. The projection of original feature space to new low dimensional space is fundamental procedure for feature extraction techniques.

Feature selection and feature extraction can be supervised or unsupervised. *Supervised* techniques result in features that are reasonable predictors of target variable. *Unsupervised* methods usually try to maintain large portion of the variability of original features in smaller number of uncorrelated new features.

Dimensionality reduction is either separated from model development or it interacts with model algorithm [1]. In the case of interaction the resulting new set of features is optimized for particular modeling technique which can make the development of different competing models difficult or impossible.

If we focus to *linear feature extraction* methods applied to document-term matrix D. Our objective is to construct matrix G that transforms terms into new features.

$$D_R = DG \tag{1}$$

Matrix D_R represents documents in a new reduced feature space. The classical example of supervised linear feature extraction is linear discriminant analysis (LDA). LDA searches for matrix G that minimizes within class variability and maximizes between class variability. Unfortunately LDA is not suitable for so large number of features as our document-term matrix has and must be wrapped in additional feature selection algorithm.

Probably the most popular feature extraction technique for document-term matrix is singular value decomposition (SVD) [3, 4, 5, 6]. SVD is unsupervised and document-term matrix D is decomposed as

$$D = VSU^T \quad (2)$$

Columns of V are orthogonal eigenvectors of DD^T while the columns of U are orthogonal eigenvectors of D^TD. S is the diagonal matrix of singular values. Singular value is the square root of eigenvalue of DD^T (or D^TD). In SVD matrix G equals to US^{-1}. More often matrices U and S are reduced to only columns that correspond to the largest singular values. Applying SVD to document-term matrix has an intuitive interpretation. The new features represent latent semantic concepts that are derived from co-occurrence of terms in the document collection.

4 Supervised mapping

The goal of the supervised feature extraction method for document-term matrix is to reduce its dimensionality with the respect to maintain as much as possible available information needed for proper classification. The document-term matrix is usually rectangular with larger number of terms than number of documents. Typical size of dictionary is several thousand terms while there are several hundreds of labeled documents in the training collection. The target categories or labels form additional column or columns of document-term matrix. Categories usually represent the topics of the documents. Instead of topics categories can stand for language, sentiment, author etc. Documents can be assigned to tens of categories. Sometimes one document belongs to more than one category thus the supervised feature extraction method should cope with multicategory assignment and should take advantage of it.

		terms (words, n-grams, concepts, ...)					category
		t_1	t_2	t_3	...	t_M	
documents	d_1	w_{11}	w_{12}	w_{13}	...	w_{1M}	y_1
	d_2	w_{21}	w_{22}	w_{23}	...	w_{2M}	y_2
	⋮	⋮	⋮	⋮	...	⋮	⋮
	d_N	w_{N1}	w_{N2}	w_{N3}	...	w_{NM}	y_N

		terms (words, n-grams, concepts, ...)					categories			
		t_1	t_2	t_3	...	t_M	c_1	c_2	...	c_K
documents	d_1	w_{11}	w_{12}	w_{13}	...	w_{1M}	$i_{1(c1)}$	$i_{1(c2)}$	...	$i_{1(cK)}$
	d_2	w_{21}	w_{22}	w_{23}	...	w_{2M}	$i_{2(c1)}$	$i_{2(c2)}$	...	$i_{2(cK)}$
	⋮	⋮	⋮	⋮	...	⋮	⋮	⋮	...	⋮
	d_N	w_{N1}	w_{N2}	w_{N3}	...	w_{NM}	$i_{N(c1)}$	$i_{N(c2)}$	...	$i_{N(cK)}$

Fig. 2 Document-term matrix enhanced by single target category (left) or set of target categories (right). In the multicategory example categories are coded as numeric indicators.

We present supervised feature extraction technique which is based on two simple assumptions:

- The final class or classes are precisely known for each training document.
- The training document is typical representative for its classes thus new document can be compared with the representatives.

The method consists of two stages. The first stage reduces number of dimensions to number of training documents. The second stage continues the reduction to the number of target categories. Both stages take into account supervised nature of the training data.

In the first stage the similarity vector with all training documents is computed for each document. Thus the document is described by the similarity vector instead of former vector of terms. The number of similarities for particular document is the same as number of training documents.

To proceed to the second stage we need to convert the target category column in our document-term matrix into indicator columns. The transformation is depicted on Fig. 3. Then the output vector from the first stage is compared with each indicator column vector and similarities with categories are derived. The data processing in the second stage is the same as in the first one. In the first stage we compare inputs with training document vectors and in the second stage we compare second stage inputs with category indicator vectors. Finally we get as many similarities as number of categories. These second stage similarities are our extracted features. If the documents in training collection are labeled with more than one category the process does not need any modification.

Processing in the second stage is supported by assumptions that labeled training documents perfectly represent their categories and that they are ideal examples that must not be assigned to other categories. This assumptions are implemented by the 1/0 category. However documents do not always include same topic with the same intensity. Thus the indicators can be changed into continuous weights when computing the similarities in second stage. We propose to set the second stage category weights with respect to similarity among the documents belonging to the same category. If the training document is similar to the documents representing particular category its weight in this category should be large and vice versa. We propose to set the weight between the training document and the category proportional to the average similarity between training document and all documents labeled by the category. If the document is also labeled by the category its own similarity to itself does not influence the weight. This weight adjusting should discriminate poorly labeled training documents and promote typical representatives.

category
c_1
c_2
⋮
c_K

categories			
c_1	c_2	...	c_K
1	0	...	0
0	1	...	0
⋮	⋮	...	⋮
0	0	...	1

Fig. 3 Schema of transformation of single target category to set of numeric indicators

Both stages can be expressed by simple multiplication of matrices. No computationally intensive matrix function is necessary. Thus the process is fast and

simple to implement. Let us set the following notation:

D … document-term training matrix without target columns (NxM)

C … document-category indicator training matrix (NxK)

S … matrix of similarities among the training documents (NxN)

R … document-extracted-feature matrix for training documents (NxK)

d … row vector of new unclassified document (1xM)

s … row vector of similarities between new document and training documents (1xN)

r … row vector of extracted features for new document (1xK)

The cosine similarity measure is used in both stages. It is very common measure in document processing algorithms [9] and it is independent on document vector size. The cosine similarity measure between vectors x and y is defined as

$$s_{x,y} = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}} \tag{3}$$

Size of document vector can reflect the length of the document which is undesirable. Using cosine similarity only the angles between vectors are compared.

We can record the method of feature extraction from training matrix as

$$S = \frac{DD^T}{\left(diag\left(DD^T\right)\right)^{\frac{1}{2}}\left(\left(diag\left(DD^T\right)\right)^{\frac{1}{2}}\right)^T}$$
$$R = \frac{SC}{\left(diag\left(SS^T\right)\right)^{\frac{1}{2}}\left(\left(diag\left(C^TC\right)\right)^{\frac{1}{2}}\right)^T} \tag{4}$$

If we would like to adjust second stage binary indicators to reflect the similarities among training documents the matrix C can be substitute by product S and C. Then the weights in document-category matrix are proportional to the sum of similarities to all documents in particular category. We do not adjust sum to mean because we use the cosine similarity which does not reflect the lengths of vectors. In addition to eliminate similarities with same training document the main diagonal of S should be set to zeros before adjustment. The modified algorithm should be recorded as

$$S = \frac{DD^T}{\left(diag\left(DD^T\right)\right)^{\frac{1}{2}}\left(\left(diag\left(DD^T\right)\right)^{\frac{1}{2}}\right)^T}$$
$$R = \frac{S(S-I)C}{\left(diag\left(SS^T\right)\right)^{\frac{1}{2}}\left(\left(diag\left(((S-I)C)^T(S-I)C\right)\right)^{\frac{1}{2}}\right)^T} \tag{5}$$

Similarly we can derive the reduced set of features for new unlabeled document using supervised information in training labeled document-term matrix as

$$s = \frac{dD^T}{\left(dd^T\right)^{\frac{1}{2}}\left(\left(diag\left(DD^T\right)\right)^{\frac{1}{2}}\right)^T}$$
$$r = \frac{sC}{\left(ss^T\right)^{\frac{1}{2}}\left(\left(diag\left(C^TC\right)\right)^{\frac{1}{2}}\right)^T} \tag{6},$$

or with modification of the second stage that takes advantage of training documents similarity matrix *S* as

$$s = \frac{dD^T}{\left(dd^T\right)^{\frac{1}{2}}\left(\left(diag\left(DD^T\right)\right)^{\frac{1}{2}}\right)^T}$$
$$r = \frac{s(S-I)C}{\left(ss^T\right)^{\frac{1}{2}}\left(\left(diag\left(((S-I)C)^T(S-I)C\right)\right)^{\frac{1}{2}}\right)^T} \tag{7}$$

Symbol I stands for square identity matrix. The function diag() transforms the diagonal of matrix into column vector. Divisions as well as square roots in the above formulas are elementwise (not matrix) operations. The denominators of the fractions are just only the norms in the cosine similarity measure expressed by matrix operations. Reader can easily derive analogous formulas for extracting features from matrix of several new unlabeled documents.

The whole process of expression of new documents by extracted features is depicted on Fig. 4. The process of developing the feature extractor together with its evaluation is on

Fig. 7.

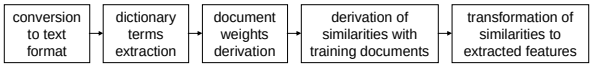

Fig. 4 steps of dimensionality reduction for unlabeled document

5 Neural network simulation

To clarify the process of extraction of small number of new highly predictive features feed-forward neural network can be used as simulator of data flow. Let us form three-layer network.

Neurons in the first input layer are receivers of document weights. There are as many input neurons as number of terms in the dictionary. Input neurons only pass the documents weights to the neurons in the second layer.

The second layer is the hidden layer. It represents the first stage of our algorithm. Each neuron corresponds

to one training document thus the number of neurons in the second layer is the same as the number of training labeled documents. The neuron computes the cosine similarity between input document and corresponding training document. If we pretend that input synaptic weights are the items in training document vector the neuron computes its potential and adjusts it by the norms of synaptic weights and input signals.

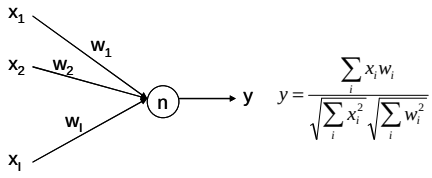

Fig. 5 neuron of hidden and output layers

The third layer is output layer and provides final extracted features. It implements the second stage of our algorithm. Neurons proceed the signals same way as neurons in second layer. Each neuron corresponds to one target category thus the number of neurons in the third layer is the same as the number of categories. If we use aforementioned binary coding of target categories each neuron from the third layer is connected only with second layer neurons that belong to represented category. Thus synaptic weights are also binary. If the modified training document-category matrix is used instead of binary indicators, the hidden and output layers are fully connected with continuous synaptic weights.

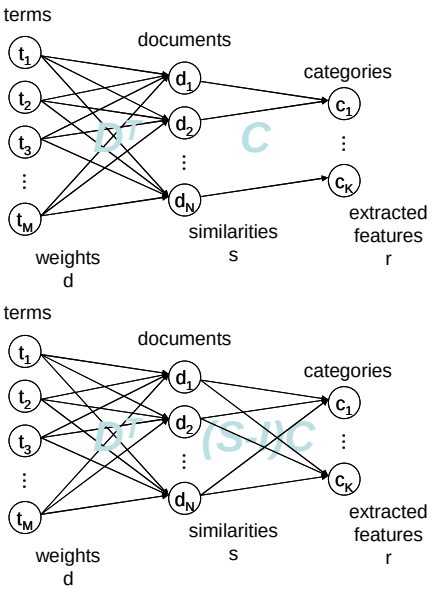

Fig. 6 topologies of the simulative network implementing supervised feature extraction

Presented feed-forward network is of fix topology. Number of neurons is derived from number of terms, documents and categories in training data. Also connections are determined by training document-term and document-category matrices.

Configuration of the network is straightforward too. No iterative process to learn the synaptic weights or other parameters is necessary. Synaptic weights are determined by the values in training document-term and document-category matrices. The transfer functions of hidden and output layers are cosine similarities.

The active usage of the built network is reasonably fast. The two feed-forward steps correspond to matrix multiplication. Output from both hidden and output layers can be useful for subsequent predictive models. Choosing between outputs from the hidden or output layer depends on the desired level of dimensionality reduction.

6 Experimental design and data

To compare described supervised feature extractor (SFX) method with the popular SVD we downloaded 645 press releases written in Czech language. The press releases were published by Czech News Agency (ČTK) or Czech publishing company Grand Prince (GP) in July 2007. All press releases are manually assigned to one of eight categories (cars, housing, travel, culture, Prague, domestic news, health, foreign news). All categories are roughly equally occupied. The typical length of the documents converted to text format is about 5kB.

The process of evaluation of the extractor is depicted on

Fig. 7. Before processing the documents were converted to text format. Then the documents were split to training (65%) and test (35%) sets randomly.

Each document was parsed to tokens. Tokens are separated by spaces in documents. Before tokens were extracted each non letter character were substituted by space and each sequence of spaces was trimmed to only one space. This easy algorithm dug out few undesirable words but they were usually eliminated by frequency filter. The dictionary included all the words in training collection that exceeded global threshold for minimal number of documents containing the word. There were 5320 words in the dictionary.

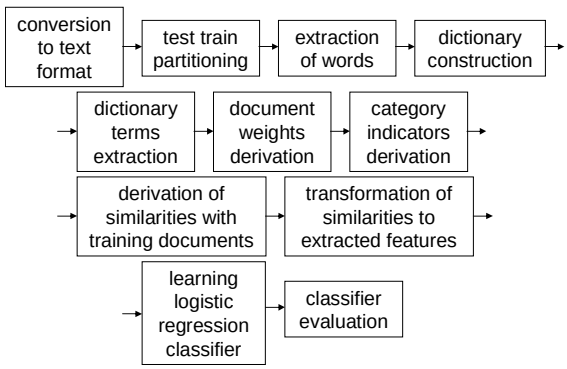

Fig. 7 process of developing and testing the supervised feature extractor (SFX)

After setting the dictionary the dictionary terms were extracted from each document in training and testing collections and the document weights were set. We used popular tf-idf weights [7, 8, 9] defined as

$$tfidf = tf \log(n / df) \quad (8),$$

tf stands for term frequency in the document, df stands for number of documents where the term is present and n is number of documents in the collection.

Documents were organized by topics into folders. We derived document-category matrix occupied by binary indicators. We did not deal with multitopic documents thus for each document just one indicator was set to one (see Fig. 3). The indicators were target variables for predictive models developed in the end of our experiment.

Transposed training document-term matrix contains synaptic weights assigned to connections between first and second layer of the network. The document-category matrix contains synaptic weights for connections between second and third layer. Or the connections between second and third layer were easily assigned using similarities among training documents and document-category indicators. Thus after extraction document-term and document-category matrices from training data the network is ready to extract features from all submitted documents.

To test the relevance of extracted features from SFX we let them enter to binomial logistic regression. For each target category we built separate classifier and measured its quality by Gini measure.

We also extracted number of features from document-term matrix same by SVD and used them in competitive logistic regression classifiers.

7 Results

Presented SFX and competitive SVD were implemented in SPSS software using its matrix syntax language. The time to extract new features for our document collection was significantly higher for SVD. To compute SVD over 645 documents and 5320 terms took 15 hours. In comparison our SFX method that uses only matrix multiplication took only 4 minutes or 6 minutes with modified second stage.

The extracted features in SVD are not optimized for supervised learning thus SFX should provide better predictors for consecutive models. The results are depiscted on Fig. 8. SFX provided significantly more useful predictors for all models. Using only binary connections between hidden and output layers we observed the tendency of some models to overfit training data. It can be caused by incorrect labeling of some training documents and more probably by partial similarities among documents assigned to different topics. The overfit was observed significantly rarely in continuous weight modification of SFX. Even the quality of subsequent model was usually better for continuous weights.

Quality of logistic regression models

		SVD		SFX (binary weights)		SFX (continuous weights)	
	Gini	train	test	train	test	train	test
topic	cars	0,93	0,98	1,00	0,95	0,99	0,98
	housing	0,89	0,76	1,00	0,96	0,95	0,91
	travel	0,55	0,51	1,00	0,30	0,86	0,71
	culture	0,76	0,60	1,00	0,89	0,99	0,97
	Prague	0,50	0,32	0,99	0,76	0,88	0,79
	domestic news	0,39	0,44	1,00	0,87	0,92	0,92
	health	0,89	0,92	1,00	0,98	0,96	0,99
	foreign news	0,08	0,25	1,00	0,94	0,92	0,95

Fig. 8 comparison of binary classifiers using different feature extraction methods

8 Conclusion and future enhancements

We described and tested simple supervised feature extraction algorithm and simulated its performance by feed-forward neural network. The experiments confirmed that extraction is reasonably fast as well as learning process. The extracted features by SFX are of high predictive potential comparing to those extracted by popular SVD.

The important characteristic of proposed supervised extraction method SFX is its ability to optimize extraction process for several target variables together. In addition it is very easy to implement SFX with different similarity measure or to change the way how the weights in document-term matrix are expressed. Even the binary coding for target categories can be changed to probabilistic coding which better expresses main and complementary topics of each document. This modification can replace our modification in the second stage.

The simulation of the extraction process by neural network enables to experiment with numbers of neurons. The number of hidden neurons is determined by number of documents in training collection. The proposed algorithm SFX can be enhanced by document (neuron) selection algorithm that picks out only typical documents for target categories and filters out the training documents that are imperfectly manually classified.

9 References

[1] Saeys Yvan, Inza Inaki, Larranaga Pedro. A review of feature selection techniques in bioinformatics. *Bioinformatics*, Vol. 23, No 19, 2007, 2507-2517

[2] Radovanovic Miloš, Ivanovic Mirjana. Text Mining: Approaches and Applications. *Novi Sad Journal of Mathematics*, Vol. 38, No. 3, 2008, 227-234

[3] Landauer Thomas, Foltz Peter, Laham Darrell. An Introduction to Latent Semantic Analysis, *Discourse Processes*, No. 25.,1998, 259-284

[4] Berry Michael, Dumais Susan. O'Brien Gavin. Using Linear Algebra for Intelligent Information Retrieval, *SIAM Rev*., Vol. 37, No. 4, 1995, 573-595

[5] Deerwester Scott, Dumais Susan, Furnas George, Landauer Thomas, Harshman Richard. Indexing by Latent Semantic Analysis, *Journal of the American Society for Information Science*, Vol. 41, 1990, 391-407

[6] Steinberger Josef, Ježek Karel. Using latent semantic analysis in text summarization and summary evaluation, *Proceedings ISIM '04*, 2004, 93-100

[7] Dumais Susan. Improving the retrieval of information from external sources, *Behavior Research Methods, Instruments and Computers*, Vol. 23, No.2, 1991, 229-236

[8] Jurafsky Daniel, Martin James H. Speech and Language Processing, 2000

[9] Masayuki Goto, Takashi Ishida, Shigeichi Hirasawa. Statistical Evaluation of Measure and Distance on Document Classification Problems in Text Mining, Proceedings of the 7th IEEE International Conference on Computer and Information Technology, 2007, 674-679