Multi-Subset Selection for Keyword Extraction and Other Prototype Search Tasks Using Feature Selection Algorithms

Somol P.
Dept. of Pattern Recognition
Inst. of Information Theory and Automation
Academy of Sciences of the Czech Republic
Pod vodárenskou věží 4, 182 08 Prague 8
somol@utia.cas.cz

Pudil P.
Faculty of Management
Prague University of Economics
Jarošovská 1117/II
377 01, Jindřichův Hradec
pudil@fm.vse.cz

Abstract

We present a framework that enables the use of traditional feature selection algorithms in a new context - for building a set of subsets of specified properties. During the course of search individual items are added/removed to/from one of the subsets in the subset system one at a time to maximize an overall criterion. Different tasks of prototype search type can be solved in this alternative way depending on suitable criterion definition. The usability of the concept is shown on keyword extraction example. Further possible applications are suggested.

1. Introduction

In feature selection (FS) the search problem of finding a subset X_d of d features from the given set Y of D measurements, $d < D$, so as to maximize an adopted criterion, $J(.)$, has been of interest for a long time. An extensive framework of search methods is now available to accomplish the task [2, 6, 8, 10]. In the following we investigate the possibility of using the existing methods for solving a different class of problems. The motivation comes from the context of document analysis [11, 12] – we needed to find terms that characterize and distinguish sufficiently for human readers the meaning of documents contained in pre-defined document classes. The solution to be presented here is, nevertheless, more general and can be modified to allow document clustering, classification as well as prototype search in other than document-processing contexts.

In Section 1.1 the basic FS problem is recalled. Section 2 decomposes FS algorithms to building blocks. Section 3 introduces "multi-subsets". Section 3.1 redefines FS algorithms for use with "multi-subsets". Sections 4 and 5 give application examples. Section 6 discusses framework properties. Section 7 concludes the paper.

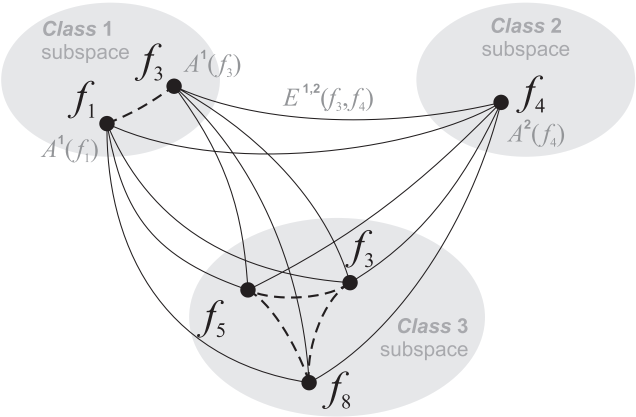

Figure 1. Relations between features in a multi-subset. Straight lines denote inter-class relations, expressed by $E(.)$ in criterion (1). Dashed lines denote "intra-class" feature importance, expressed by $A(.)$ in (1).

1.1 Basic Problem Formulation

Consider $Y = \{f_i | i = 1, \ldots, D\}$ the set of all available D measurements, and $X_d = \{f_j | j = 1, \ldots, d; f_j \in Y\}$ a subset of d features, where $d < D$ and possibly $d << D$. The goal of standard FS process is to find such subset $\tilde{X}_d$, for which the value of an adopted criterion, typically some class-distance measure (cf. [15]), $J(\tilde{X}_d)$, is maximum:

$$J(\tilde{X}_d) = \max_{X \subset Y, |X|=d} J(X)$$

In context of pattern recognition the dimensionality of the problem as a whole is reduced to d. In case of classification problems this should lead to better classification performance, in case of modelling this should result in more accurate representation of data or at least in savings in data acquisition and processing cost. One common subset of features is selected for the problem as a whole.

2. Sequential Search Abstracted

Upon closer examination, most of the known sequential feature selection algorithms can be identified to share the same "core mechanism" of adding and removing features. Let us abstract the respective algorithm steps as follows (for the sake of simplicity we consider only non-generalized algorithms which process one feature at a time only):

Definition 2.1 *Let ADD(X_d) be the operation of adding such feature f^+ to the working set X_d to obtain X_{d+1}, that*

$$f^+ = \arg \max_{f \in Y \setminus X_d} J(X_d \cup \{f\})$$

Definition 2.2 *Let REMOVE(X_d) be the operation of removing such feature f^- from the working set X_d to obtain set X_{d-1}, that*

$$f^- = \arg \max_{f \in X_d} J(X_d \setminus \{f\})$$

Using these abstracted steps it is now possible to outline the basic idea behind standard feature selection algorithms very simply. For instance:

SFS (*Sequential Forward Selection* [2]):

1. Starting with empty set X, repeat ADD(X) d times to finally get a subset of d features.

SFFS (*Sequential Forward Floating Selection* [8]):

1. Start with an empty set. $d = 0$.
2. ADD(X_d). $d = d + 1$.
3. Repeat REMOVE(X_d), $d = d - 1$ as long as it improves solutions already known for the lower d and $d > 1$.
4. If $d < D$ go to 2.

OS (*Oscillating Search* [10]):

1. Start with an initial set of d features.
2. ADD(X_d). REMOVE(X_{d+1}).
3. REMOVE(X_d). ADD(X_{d-1}).
4. If a better subset has been found, go to 2.

3. Multi-Subset Search

Assume our task is to select more than one feature subset at once while taking into account the relations between features in each subset as well as between subsets. Let us denote such system of subsets as follows:

Definition 3.1 *Let C be the number of classes or clusters to be represented by different subsets. Let $\mathbf{X}_d$ be a system of subsets, to be called a "multi-subset", where C_m depicts the size of m-th subset.*

$$\mathbf{X}_d = \{\mathcal{S}_m | m = 1, \ldots, C\}$$

where

$$\mathcal{S}_m = \{f_i^m | i = 1, \ldots, C_m; f_i^m \in Y\}$$

To capture the meaning and quality of features in a multi-subset we define an overall abstract criterion $\bar{J}(.)$, that combines two components. Denote $E^{m,n}(f_i, f_j)$ the component that is to describe the *inter-subset* relation between features $f_i \in \mathcal{S}_m$ and $f_j \in \mathcal{S}_n$, $m \neq n$. Denote $A^m(f)$ the "weight" component that is to describe the *intra-subset* importance of feature f within $\mathcal{S}_m$. Assuming both $A(.)$ and $E(.)$ can be defined reasonably for a given problem as real functions, $\bar{J}(.)$ can be defined as follows.

Definition 3.2 *Let $\bar{J}(.)$ be a criterion to describe the quality of multi-subset $\mathbf{X}_d$:*

$$\bar{J}(\mathbf{X}_d) = \sum_{\substack{m=1,\ldots,C \\ i=1,\ldots,C_m}} A^m(f_i) \sum_{\substack{n=1,\ldots,C \\ n \neq m \\ j=1,\ldots,C_n}} E^{m,n}(f_i, f_j) \tag{1}$$

The meaning of formula (1) is illustrated in Figure 1. The concrete form of $A(.)$ and $E(.)$ depends on the problem to be solved and will be illustrated by example in Section 4.

Remark: we say that the size of the multi-subset $\mathbf{X}_d$ is $d = \sum_{m=1}^{C} |\mathcal{S}_m|$.

3.1 Sequential Multi-Subset Search

Let us now reformulate our task using the above definitions. We need to find such system (multi-subset) $\mathbf{X}_d$ of d features, for which $\bar{J}(\mathbf{X}_d)$ is maximum. For this purpose we can easily adapt the standard feature selection algorithms as described in Sect. 2. We only need to redefine the commonly shared operations ADD() and REMOVE().

Definition 3.3 *Let ADD($\mathbf{X}_d$) be the operation of adding one feature to the working multi-subset $\mathbf{X}_d$ to obtain such multi-subset $\mathbf{X}_{d+1}$ for which*

$$\bar{J}(\mathbf{X}_{d+1}) = \max_{\substack{i=1,\ldots,C \\ f \in Y \setminus \mathcal{S}_i}} \bar{J}(\{\mathcal{S}_1, .., \mathcal{S}_{i-1}, \mathcal{S}_i \cup \{f\}, \mathcal{S}_{i+1}, .., \mathcal{S}_C\})$$

Definition 3.4 *Let REMOVE($\mathbf{X}_d$) be the operation of removing one feature from the working multi-subset $\mathbf{X}_d$ to obtain such multi-subset $\mathbf{X}_{d-1}$ for which*

$$\bar{J}(\mathbf{X}_{d-1}) = \max_{\substack{i=1,\ldots,C \\ f \in \mathcal{S}_i}} \bar{J}(\{\mathcal{S}_1, .., \mathcal{S}_{i-1}, \mathcal{S}_i \setminus \{f\}, \mathcal{S}_{i+1}, .., \mathcal{S}_C\})$$

Now any of the algorithms discussed in Sect. 2, can be used to construct multi-subset $\mathbf{X}_d$ of required size d.

4. Keyword Extraction Example

The main application field of multi-subset search is prototype search in various forms. To give a concrete example we will show two document characterization experiments. (For an overview of related issues cf., e.g., [12, 11, 13, 14].)

First, let us suggest the problem. Suppose we have received several boxes full of unknown documents. We do not want to read the documents but we want to get some idea about box contents, in particular how the boxes differ from each other. This is where multi-subset search can be applied. Note, that instead of features we will now select terms, so that the selection would give us a clue about the document contents and particularly about the main topics being characteristic for each class.

We examine two datasets representing classes of documents described by term frequency tables. Let T be the set of all terms, U^m be the set of documents in the m-th class. Let $\mathbf{F}^m = \{\mathbf{f}^m_{u,t} | u \in U^m; t \in T; \mathbf{f}^m_{u,t} \in \{0,1\}\}$ be a binary frequency table describing term presence (columns) in documents (rows) of the m-th class. *Dataset 1* represents Reuters articles grouped in 9 classes of 53, 2, 2, 61, 30, 67, 3, 127 and 63 documents, respectively, containing altogehther 3822 different terms (cf.http:// www.daviddlewis.com/ resources/). *Dataset 2* represents a selection of documents from MIU, Cambridge University, grouped in 8 classes of 86, 38, 65, 66, 35, 43, 69 and 70 documents, respectively, containing altogether 38503 different terms [3, 4]. The datasets have been pre-processed by means of stemming and stop word elimination. For this example we will use a simple, "naïve" way of term importance evaluation.

4.1 Selecting the Terms

First we need to concretize the forms of $A(.)$ and $E(.)$ to give meaning to criterion (1). Inspired by the idea of TF IDF [9], [7] we want to identify such terms, that: 1) are typical for some classes but rare in others, 2) complement each other within a class to represent as many various documents in a class as possible, 3) are not singular and 4) are not too frequent while equally distributed across too many classes. The proposed $E(.)$ promotes pairs of terms from which each one is frequent in one class but rare in the other class:

$$E^{m,n}(f_1, f_2) = (\sum_{\substack{u \in U^m \\ \mathbf{f}^m_{u,f_2}=0}} \mathbf{f}^m_{u,f_1})(\sum_{\substack{u \in U^n \\ \mathbf{f}^n_{u,f_1}=0}} \mathbf{f}^n_{u,f_2}) \qquad (2)$$

The proposed $A(.)$ should prevent overlapping of terms within the documents of one class:

$$A^m(f) = \frac{\sum_{u \in U^m} \mathbf{f}^m_{u,f}}{1 + \sum_{u \in U^m} \sum_{\bar{f} \in \mathcal{S}_m \setminus \{f\}} \mathbf{f}^m_{u,f} \mathbf{f}^m_{u,\bar{f}}} \qquad (3)$$

Figures 2 and 3 show results obtained using the SFS procedure. Coefficients show the proportion of documents in respective class that contains the term. On a P4-3Ghz CPU the first 15 steps took $<$ 1min for *dataset 1* and $\sim$ 2min for *dataset 2*. The next 15 steps took $\sim$ 2min for *dataset 1* and $\sim$ 15min for *dataset 2*.

```
Multisubset after 15 SFS steps, Crit: 63.2619
1: aluminium(0.97)
2: australian(0.79), citibank(0.98), warrant(0.94)
3: eep(1), malt(1), barley(0.99)
4: deficit(0.99), account(0.55)
5: beef(1)
6: cocoa(0.92)
7: coconut(1), philippin(0.95)
8: coffe(0.92)
9: copper(0.98)

the same after another 15 SFS steps, Crit: 133.159
1: aluminium(0.97), compani(0.43)
2: australian(0.79), citibank(0.98), warrant(0.94),
    mark(0.8), sell(0.68)
3: eep(1), malt(1), barley(0.99), usda(0.78),
    colombia(0.78), offer(0.55)
4: deficit(0.99), account(0.55), current(0.37)
5: beef(1), unit(0.43), food(0.84)
6: cocoa(0.92), buffer(0.99), icco(1)
7: coconut(1), philippin(0.95), product(0.43)
8: coffe(0.92), quota(0.85), intern(0.39)
9: copper(0.98), mine(0.86)
```

Figure 2. Dataset 1: extracted keywords

```
Multisubset after 15 SFS steps, Crit: 26.0438
1: collabor(0.28),
2: content(0.26), reader(0.39)
3: simul(0.29), electr(0.49)
4: commerc(0.4)
5: patient(0.73), medic(0.69), clinic(0.83)
6: intellig(0.26), represent(0.21)
7: busi(0.2), competit(0.29)
8: send(0.26), protocol(0.3)

the same after another 15 SFS steps, Crit: 60.3019
1: collabor(0.28), cours(0.23), social(0.24)
2: content(0.26), reader(0.39),
    topic(0.28), digit(0.3)
3: simul(0.29), electr(0.49), paramet(0.28)
4: commerc(0.4), compani(0.19), transact(0.2)
5: patient(0.73), medic(0.69), clinic(0.83),
    physician(0.92), hospit(0.78), health(0.59)
6: intellig(0.26), represent(0.21), algorithm(0.22)
7: busi(0.2), competit(0.29), firm(0.35),
    market(0.24)
8: send(0.26), protocol(0.3), architectur(0.24),
    tion(0.26)
```

Figure 3. Dataset 2: extracted keywords

It should be emphasized that the meaning of the results depends completely on the adopted criterion, which in this case is not constructed for classifier optimization, but only for pointing out class-significant topics to a human reader.

Remark: Even with the $E(.)$ and $A(.)$ defined as (2) and (3) the resulting term multi-subsets can be used as naïve classifiers. An unknown document can be classified to m-th class based on significant presence of terms from $\mathcal{S}_m$ (Exact definition is beyond the scope of this paper). However, the classification rate of such classifier should not be expected too good for obvious reasons.

5 Further Application Domains

The multi-subset search framework opens up the possibility to use standard feature selection algorithms for a variety of problems, including prototype search, various forms of document analysis, clustering, detection of communities in graphs, etc. It can also be used for k-NN classifier optimization. In such a case we would consider selecting patterns instead of features. Only those patterns important to define the decision boundary need to be preserved. For an overview of this problem domain see, e.g., [1, 5].

The importance of patterns in k-NN can be described by means of $E(.)$ and $A(.)$ specific definition, similarly as before in this paper. Consider redefining $E(.)$ to prefer close pairs of patterns from different classes, where for each one in the pair there is enough neighbours of its own class among its k closest neighbours (to avoid outliers). The "weight" component $A(.)$ can then be used to prevent selection of tight groups of similar patterns by taking into account the distance to neighbours in the same class.

The idea of k-NN optimization by means of multi-subset search will be investigated in our future work.

6. Multi-Subset Framework Properties

Computational complexity of the proposed framework depends strongly on the concrete form of criterion (1), but is to be expected at least C times higher than the cost of the original feature selection algorithm. Multi-subset search is a combinatiorial problem. Each feature added to the system increases the number of combinations to be evaluated in further stages. The framework is thus applicable only to selecting relatively small number of features/terms/prototypes (typically hundreds).

An important property of the proposed framework is its ability to accomodate changes in the number of subsets in a multi-subset. It is possible to add an empty subset or remove one of existing subsets and use the Oscillating Search to re-optimize the new multi-subset without the need to start from scratch. This multi-subset property opens up further applicational fields to be investigated in our future work.

Note: The presented framework definition is not intended to be strict. Instead of redefining (2) and (3) there is always the choice to redefine criterion (1) as a whole.

7. Conclusion and Future Work

We have identified common building blocks in standard feature selection algorithms. By means of redefining these blocks we have broadened the application domain of such standard – well performing – feature selection algorithms to include prototype search type of problems.

We have demonstrated the proposed multi-subset search idea on keyword extraction example. Applicability to many problems is yet to be investigated; this includes problems of document (and other) clustering, detection of communities in graphs, k-NN classifier optimization etc.

Acknowledgement: We thank Charles Fuka and Rudolf Hanka for help with the datasets. The work has been supported by the grant of the Czech Ministry of Education 1M0572 DAR and partially by GACR No. 402/03/1310 and EC project No. FP6-507752 MUSCLE.

References

[1] V. Cerverón and F. J. Ferri. Another Move Toward the Minimum Consistent Subset: A Tabu Search Approach to the Condensed Nearest Neighbor Rule *IEEE Trans. on Syst., Man, and Cybernetics*, 31, 3, JUNE 2001

[2] P. A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach*. Prentice-Hall, 1982.

[3] K. Fuka and R. Hanka. Feature Set Reduction for Document Classification Problems. In: Proc. of IJCAI-01 Workshop: Text Learning: Beyond Supervision, Seattle, 2001.

[4] K. Fuka. Distributed Knowledge Management in Digital Libraries. PhD thesis, Cambridge University, 2002.

[5] P. E. Hart. The condensed nearest neighbor rule. *IEEE Trans. Information Theory*, 14, 5, 515–516, May 1968.

[6] A. K. Jain and D. Zongker. Feature selection: Evaluation, application and small sample performance. *IEEE Transactions on PAMI*, 19:153 – 158, 1997.

[7] T. Joachims. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. Proc. Int. Conf. on Machine Learning (ICML), 1997.

[8] P. Pudil, J. Novovičová and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15:1119–1125, November 1994.

[9] G. Salton and M.J. McGill. Introduction to Modern Information Retrieval. McGraw-Hill, 1983.

[10] P. Somol and P. Pudil, Oscillating search algorithms for feature selection. In Proc. 15th ICPR, Barcelona, 406–409, 2000.

[11] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1,1–47, 2002.

[12] G.C. Stein, A. Bagga and G. Bowden Wise. Multi-Document Summarization: Methodologies and Evaluations. In: Proc. TALN 2000, Lausanne, 2000.

[13] B. Stein and S. Meyer zu Eissen. Automatic Document Categorization Interpreting the Perfomance of Clustering Algorithms *LNAI 2821*, 254–266, Springer, 2003.

[14] P. D. Turney. Learning Algorithms for Keyphrase Extraction. *Information Retrieval*, 2:4, 303–336, Kluwer, 2000.

[15] A. Webb, *Statistical Pattern Recognition*, 2nd Ed., John Wiley & Sons, 2002.