Composite Kernels for Hypertext Categorisation

Thorsten Joachims THORSTEN.JOACHIMS@GMD.DE
GMD - German National Research Center for Information Technology, Institute for Autonomous Intelligent Systems, Team Knowledge Discovery, Schloss Birlinghoven, 53754 Bonn, Germany

Nello Cristianini NELLO@DCS.RHBNC.AC.UK
John Shawe-Taylor JOHN@DCS.RHBNC.AC.UK
Department of Computer Science, Royal Holloway, University of London, Egham, Surrey TW20 0EX, UK

Abstract

Kernels are problem-specific functions that act as an interface between the learning system and the data. While it is well-known when the combination of two kernels is again a valid kernel, it is an open question if the resulting kernel will perform well. In particular, in which situations can a combination of kernel be expected to perform better than its components considered separately? We investigate this problem by looking at the task of designing kernels for hypertext classification, where both words and links information can be exploited. We provide sufficient conditions that indicate when an improvement can be expected, highlighting and formalising the notion of "independent kernels". Experimental results confirm the predictions of the theory in the hypertext domain.

1. Introduction

Kernel-Based learning methods (KMs) are a general class of pattern recognition algorithms that work by first implicitly mapping the data into a high-dimensional feature space, and subsequently executing a simple learning algorithm (e.g. linear separation; regression; clustering; PCA). Their modularity makes it possible to separately study the problem of designing good feature mappings and the problem of designing the learning algorithm. In this paper we will focus on the first problem. A good feature representation – in this case of hypertext documents – can the be used with any kernel-based learning algorithm (see Cristianini and Shawe-Taylor (2000) for some examples).

KMs present a unique property from the software engineering point of view: many of the typical design choices are reduced to the choice of a suitable kernel function. Since kernel functions can be regarded as (special) similarity measures between inputs, one can (and should) use domain knowledge to design them. In doing so, one should always make sure that the similarity function being used is a 'valid' kernel, that is that it can be rewritten as an inner product $K(x,z) = \langle \phi(x), \phi(z) \rangle$ for some $\phi : X \to \mathcal{F}$ defining a feature space. A useful method for designing complex kernels is to start from simple and well understood ones, and then transform or combine them by a series of 'kernel preserving' operations, hence constructing an increasingly matching feature space. This technique is very promising in fields in which there is no real understanding of a good similarity measure between data, but several independent approaches seem to perform well (e.g. in image retrieval one can retrieve by color, texture, shapes, etc.).

Since in general the presence of irrelevant features decreases the performance of KMs, one should be able to assess a priori which kernel combinations are likely to generalise well. Our goal is to characterise situations where a combination of kernel can be expected to perform better than its components considered separately. Intuitively, one would like each of the two kernels to contribute information that is not available to the other. While others have already formulated and empirically evaluated similar postulates for other learning methods (e.g. Ali and Pazzani (1996), Ng and Kantor (1998)), there is no formal characterisation linking all influence factors to a performance improvement of the combined learner. Such a theory must consider the data at hand, both kernels, and also the task in form of the information given by the labels. The theoretical analysis we provide gives a first sufficient condition for error reduction by kernel combination, and our experiments confirm the predictions.

The application studied in this paper is the case of combining two different feature representations of web-pages, one based on link information, the other based on words. In particular, we introduce a new type of kernel whose Gram matrix is the co-citation matrix known in bibliometrics. We show how such a matrix can be combined with the document-by-document similarity matrix known in information-retrieval to obtain a valid kernel that performs better than the other two considered separately.

The idea of using link structure information to improve retrieval of hypertext documents is at the basis of some very successful algorithms, such as PageRank (Page & Brin, 1998) and HITS (Kleinberg, 1999). Such information can (and should) of course be fused with the information about the text of the document so that it captures as much relevant information as possible from the corpus. This problem has been studied for example by Cohn and Hofmann (2000) in the context of probabilistic generative models, by Craven et al. (1998) for exploiting relations between web pages, and by Blum and Mitchell (1998) in the task of classification using unlabeled documents.

2. Kernel Methods

Kernel methods are a new approach to solving machine learning problems. By developing algorithms that only make use of inner products between images of different inputs in a feature space $\mathcal{F}$, they can be applied in very rich feature spaces provided the inner products can be computed implicitly. In this way they avoid explicitly computing the feature vector for a given input. One of the key advantages of this approach is its modularity: the decoupling of algorithm design and statistical analysis from the problem of creating appropriate function/feature spaces for a particular application. Furthermore, the design of kernels themselves can be performed in a modular fashion: simple rules exist to combine or adapt basic kernels to construct more complex kernels that are guaranteed to again correspond to an inner product in some space (see Section 4).

Given a training set $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$, the information available to kernel based algorithms is contained entirely in the matrix of inner products

$$G = K = (\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^{m},$$

known as the Gram or kernel matrix. Kernels can be used without actually knowing the feature space $\mathcal{F}$, as long as one can guarantee that *some* such space exists and that the kernel can be regarded as an inner product in that space. One can characterise valid kernel functions in many ways. The simplest way is probably the following:

Proposition 1 *(Saitoh, 1988) A function $\mathcal{K}(x, z)$ is a valid kernel iff for any finite set it produces symmetric and positive definite Gram matrices.*

The solutions sought by kernel-machines are linear functions in the feature space $f(\mathbf{x}) = \mathbf{w}'\mathbf{x}$, for some weight vector $\mathbf{w}$, where $'$ denotes the transpose of a vector or matrix. The kernel trick can be applied whenever the weight vector can be expressed as a linear combination of the training points, $\mathbf{w} = \sum_{i=1}^{m} \alpha_i \phi(\mathbf{x}_i)$, implying that we can express f as follows

$$f(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}).$$

2.1 Kernels for Text

The bag-of-words (BOW) (or bag-of-terms) representation reduces a document to a histogram of word frequencies. It has a natural representation as a vector, where each entry of the vector is indexed by a specific term, and the components of the vector are the frequency of the term in the given document. Typically such a vector is then mapped into some other space, where the word frequency information is merged with other information (e.g. word importance, where uninformative words are given low or no weight).

In this way a document is represented by a (column) vector d. Typically d can have tens of thousands of entries, often more than the number of documents. However, for a particular document the representation is extremely sparse, having only relatively few non-zero entries. A corpus is represented by a matrix D, whose columns are indexed by the documents and whose rows are indexed by the terms, $D = [d_1, ..., d_m]$. We also call the data matrix D the "term by document" matrix. We define the "document by document" matrix to be $G = D'D$; the "term by term" matrix to be $T = DD'$.

If we consider the feature space defined by the basic vector-space model, the corresponding kernel is given by the inner product between the feature vectors

$$\mathcal{K}(d_1, d_2) = \langle d_1, d_2 \rangle = d_1' d_2,$$

In this case the Gram matrix is just the document by document matrix, well known in Information Retrieval.

2.2 Co-citation Kernels

Consider the co-citation matrix of a set of documents. This concept was first introduced in the context of bibliometrics to analyse impact factors or to detect clusters of related documents. Two documents have a positive score if they are cited by the same document. It is natural to extend this idea to a co-link matrix for hypertext documents, possibly also considering out-links as well as in-links. Chang et al. (2000) used the co-citation matrix for a problem of retrieval in citation analysis and Chakrabarti et al. (1998) exploited similar information in their classification model. Furthermore, one could define link-weighting schemes in a similar way to the term-weighting schemes used in text categorisation, where some links are more informative than others.

It is easy to see that this co-citation matrix is also a Gram Matrix: the feature space generated by this kernel has as many dimensions as the number of documents (or twice as many) if one considers only one direction (both directions) for the links. The document representation implied by this Gram matrix will be called "bag-of-links" (BOL). The underlying assumption for using this kernel is that documents on a similar topic will be linked by (link to) approximately the same documents. Experiments reported in Section 5 confirm this expectation.

We finish this section by giving a formal definition of the link features that we will be using.

Definition 1 *Let $(\mathcal{D}, \mathcal{E})$ be a hypertext collection, where $\mathcal{D}$ is a set of documents and $\mathcal{E} \subset \mathcal{D} \times \mathcal{D}$ is a set of links, (d_1, d_2) indicating that document d_1 contains a link to document d_2.*

The inlink *feature map ϕ_ι is given by*

$$\phi_\iota(d_1)_{d_2} = \begin{cases} 1 & \text{if} (d_2, d_1) \in \mathcal{E}; \\ 0 & \text{otherwise.} \end{cases}$$

Similarly, the outlink feature vector ϕ_o is related to ϕ_ι by $\phi_o(d_1)_{d_2} = \phi_\iota(d_2)_{d_1}$.

3. On Kernel Combination

Closure properties for the class of kernel functions can easily be deduced from their mathematical characterisation in terms of the positive-definiteness of the resulting Gram matrix. The following proposition can be viewed as showing that kernels satisfy a number of closure properties, allowing us to create more complicated kernels from simple building blocks.

Proposition 2 *Let $\mathcal{K}_1$ and $\mathcal{K}_2$ be kernels over $X \times X$, $X \subseteq \mathbb{R}^n$, $a \in \mathbb{R}^+$, $0 \leq \lambda \leq 1$, $f(\cdot)$ a real-valued function on X, $\phi : \mathbf{X} \longrightarrow \mathbb{R}^\mathbf{m}$ with $\mathcal{K}_3$ a kernel over $\mathbb{R}^m \times \mathbb{R}^m$, and $\mathbf{K}$ a symmetric positive semi-definite $n \times n$ matrix. Then the following functions are kernels:*

$$\begin{aligned}
\mathcal{K}(\mathbf{x}, \mathbf{z}) &= \lambda \mathcal{K}_1(\mathbf{x}, \mathbf{z}) + (1-\lambda)\mathcal{K}_2(\mathbf{x}, \mathbf{z}), \\
\mathcal{K}(\mathbf{x}, \mathbf{z}) &= a\mathcal{K}_1(\mathbf{x}, \mathbf{z}), \\
\mathcal{K}(\mathbf{x}, \mathbf{z}) &= \mathcal{K}_1(\mathbf{x}, \mathbf{z})\mathcal{K}_2(\mathbf{x}, \mathbf{z}), \\
\mathcal{K}(\mathbf{x}, \mathbf{z}) &= f(\mathbf{x})f(\mathbf{z}), \\
\mathcal{K}(\mathbf{x}, \mathbf{z}) &= \mathcal{K}_3(\phi(\mathbf{x}), \phi(\mathbf{z})), \\
\mathcal{K}(\mathbf{x}, \mathbf{z}) &= \mathbf{x}'\mathbf{K}\mathbf{z}.
\end{aligned}$$

The proof of all but the first of these results is given by Cristianini and Shawe-Taylor (2000). It is easy to see that convex combinations of kernels yield valid kernels (since the spectrum of resulting Gram matrices would always be positive). We will, however, give an explicit construction of the corresponding feature space since it will prove useful later.

Let $\phi^i : X \to \mathbb{R}^{m_i}$ be the feature mapping corresponding to the kernel $\mathcal{K}_i$, $i = 1, 2$. Now consider the feature vector

$$\phi(x) = [\sqrt{\lambda}\phi^1(x), \sqrt{(1-\lambda)}\phi^2(x)].$$

The corresponding inner product satisfies

$$\begin{aligned}
\langle \phi(x), \phi(z) \rangle &= \lambda \langle \phi^1(x), \phi^1(z) \rangle + (1-\lambda)\langle \phi^2(x), \phi^2(z) \rangle \\
&= \lambda \mathcal{K}_1(x, z) + (1-\lambda)\mathcal{K}_2(x, z) \\
&= \mathcal{K}(x, z)
\end{aligned}$$

giving an explicit construction of the feature space corresponding to the (therefore) valid kernel $\mathcal{K}$.

In the following, we present a theory aimed at capturing the notion that combining two kernels that separately perform equally well is going to help as long as the kernels are 'independent', that is they do not extract the same features. This is to some extent similar to boosting, where combining independent hypotheses can generate a better overall hypothesis.

There are two main ways to justify a claim of 'better generalisation'. The first is to provide experimental evidence for improvements on realistic datasets. We will show such experiments in the next section. The second way is to show that the method improves the quality of a rigorous bound on the generalisation error. We will describe conditions under which such an improvement can be expected when two kernels are combined. The conditions are validated in experiments showing that observed improvements coincide with the situations when the theory predicts that improvement should occur.

Our approach is to consider a particular kernel method, namely the soft-margin Support Vector Machine (SVM) (Cortes & Vapnik, 1995). For the SVM bounds on the generalisation error are known and we will analyse when such bounds improve with kernel combination. The bounds we will use are the so-called soft-margin generalisation error bounds. Bounds of this type have been derived in several different frameworks and hence appear to capture a key property of the learned function. Indeed, the soft-margin Support Vector Machine (SVM) denoted by h_{SVM} below optimises a quantity closely related to bounds of this type. For a given training set $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_m)$, $y_i \in \{-1, +1\}$ of m examples the SVM solves the following quadratic program.

$$\begin{aligned} \text{min:} \quad & V(\mathbf{w}, b, \xi) = \frac{1}{2}\,\mathbf{w}'\mathbf{w} + C\,\sum_{i=1}^{m} \xi_i \\ \text{s.t.:} \quad & \forall_{i=1}^{m} : y_i[\mathbf{w}'\phi(\mathbf{x}_i) + b] \geq 1 - \xi_i \\ & \forall_{i=1}^{m} : \xi_i > 0 \end{aligned}$$

The following two bounds relate the solution of this optimisation problem to the error of the SVM. The first is given in the expected error framework. We denote with $\gamma_i = y_i\,[\mathbf{w}'\phi(\mathbf{x}_i) + b] = y_i[\sum_{j=1}^{m} \alpha_j y_j \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) + b]$ the margin of a training example.

Theorem 1 *(Joachims, 2001)* **Bound on Expected Error of Soft-Margin SVMs** *The expected error rate $\mathcal{E}(Err^m(h_{SVM}))$ of a soft-margin SVM based on m training examples with $c \leq \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) \leq c + R^2$ for all points with non-zero probability and some constant c, is bounded by*

$$\mathcal{E}(Err^m(h_{SVM})) \leq \frac{\rho\,\mathcal{E}(R^2\mathbf{w}^2) + \rho\,C'\,\mathcal{E}\left(\sum_{i=1}^{m+1} \xi_i\right)}{m+1}$$

with $C' = C\,R^2$ if $C \geq 1/(\rho\,R^2)$, and $C' = C\,R^2 + 1$ otherwise. For unbiased hyperplanes ρ equals 1, and for stable hyperplanes ρ equals 2. The expectations on the right are over training sets of size $m+1$.

The second theorem we will quote is proven in the statistical learning theory framework but the form of the bound is very similar to that given in Theorem 1, once the standard bound on the fat shattering dimension of linear functions is substituted.

Theorem 2 *(Shawe-Taylor & Cristianini, 2000) Let X be a sturdy class of real-valued functions with range $[-a, a]$ and fat-shattering dimension bounded by $\mathrm{fat}_X(\gamma)$. Fix a scaling of the output range $\kappa \in \mathbb{R}^+$. Consider a fixed but unknown probability distribution on the input space X. Then with probability $1 - \delta$ over randomly drawn training sets S of size m for all $a > \gamma > 0$ the generalisation of a function $\mathrm{sign}(f) \in \mathrm{sign}(X)$ is bounded by*

$$\begin{aligned} \epsilon(m, d_1, d_2, \delta) \quad = \quad & \frac{2}{m}\left(d_1 \log_2\left(256m\left(\frac{a}{\gamma}\right)^2\right)\right. \\ & \log_2\left(\frac{16ema}{\gamma}\right) + d_2 \log_2(2em) \\ & \left. + \log_2\left(\frac{8m^2a}{\delta\kappa}\right)\right), \end{aligned}$$

where

$$d_1 = \mathrm{fat}_X(\gamma^-/16), \quad \text{and} \quad d_2 = \left\lfloor \frac{2(D'(S, f, \gamma) + \kappa)}{\gamma} \right\rfloor$$

provided $m \geq 2/\epsilon$ and there is no discrete probability on misclassified training points.

Taken together Theorems 1 and 2 indicate that the key quantity in estimating the generalisation is the soft margin

$$B = \frac{R^2}{\delta^2} + C' \sum \xi_i, \tag{1}$$

where R is the radius of the ball containing the support of the distribution, C' is a constant and ξ_i are the slack variables relative to a margin $\delta^2 = 1/\mathbf{w}^2$. We will therefore take equation (1) as the basis for our analysis of the performance obtained by combining kernels. If combining two kernels leads to a lower value of B, then combining the two optimises an upper bound.

Definition 2 SV Redundancy on a Training Example: *Let $\gamma_{a,i}$ and $\gamma_{b,i}$ be the margins for training example i of two SVMs trained with different kernels $\mathcal{K}_a$ and $\mathcal{K}_b$ on the same training sample. Then*

$$\Delta_i(\lambda, \Gamma) = [1 - \frac{1}{\Gamma}(\lambda\gamma_{a,i} - (1-\lambda)\frac{1}{2}\gamma_{b,i}]_0$$

is their Support Vector Redundancy on training example i at margin Γ. $[.]_0$ is the identity if the argument is positive, otherwise it is 0.

Definition 3 SV Redundancy on Training Set: *Let Δ_a and Δ_b be the vectors of SV Redundancies of two SVMs trained with different kernels $\mathcal{K}_a$ and $\mathcal{K}_b$ on the same training sample. Then*

$$\Delta(\lambda, \Gamma) = \sum_{i}^{m} \Delta_i(\lambda, \Gamma)$$

is their SV Redundancy at margin Γ on the whole training sample.

Theorem 3 Bound on the Soft Margin of Combined Kernel: *Let $[\mathbf{w}_a, b_a]$ and $[\mathbf{w}_b, b_b]$ be the solutions of two SVMs trained with different kernels $\mathcal{K}_a$ and $\mathcal{K}_b$ on the same training sample. We will assume that the radius of the ball containing the data is centred around the origin. Then the value of B of an SVM with combined kernel $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \lambda\mathcal{K}_a(\mathbf{x}_i, \mathbf{x}_j) + (1-\lambda)\mathcal{K}_b(\mathbf{x}_i, \mathbf{x}_j)$ is less than*

$$\frac{\lambda}{\Gamma^2}\frac{R_a^2}{\delta_a^2} + \frac{(1-\lambda)}{\Gamma^2}\frac{R_b^2}{\delta_b^2} + 2C'\Delta(\lambda, \Gamma)$$

for an appropriate value of $C' \geq 0$ and any $\Gamma \geq 0$.

Proof *The value of B of the combined classifier can be upper bounded using the solution of the soft-margin SVM training problem:*

$$\frac{R^2}{\delta^2} + CR^2\sum_{i=0}^{m}\xi_i$$
$$\leq 2R^2 min_{\mathbf{w},b}\frac{1}{2}\mathbf{w}'\mathbf{w} + C\sum_{i=0}^{m}[1 - y_i(\mathbf{w}'\phi(\mathbf{x}_i) + b)]_0$$

If R^2 can be upper bounded, then any value for $\mathbf{w}$ and b will be an upper bound for the soft margin of the combined classifier. Let ϕ^a be the feature vector for the first representation and ϕ^b that for the second. We form the feature vector $\phi_\lambda(\mathbf{x}) = \left[\sqrt{\lambda}\phi^a, \sqrt{1-\lambda}\phi^b\right]$, and use the hyperplane

$$\mathbf{w}(\lambda, \Gamma) = \left[\frac{\sqrt{\lambda}}{\Gamma}\mathbf{w}_a, \frac{\sqrt{1-\lambda}}{\Gamma}\mathbf{w}_b\right], b = \lambda b_a + (1-\lambda)b_b.$$

The square norm of the weight vector is

$$\begin{aligned}\mathbf{w}(\lambda, \Gamma)^2 &= \frac{\lambda}{\Gamma^2}\mathbf{w}_a\prime\mathbf{w}_a + \frac{1-\lambda}{\Gamma^2}\mathbf{w}_b\prime\mathbf{w}_b \\ &= \frac{\lambda}{\Gamma^2\delta_a^2} + \frac{1-\lambda}{\Gamma^2\delta_b^2}\end{aligned}$$

and an upper bound for the maximum square norm of the combined feature vectors is

$$\begin{aligned} & max_i \lambda\phi^a(\mathbf{x}_i)'\phi^a(\mathbf{x}_i) + (1-\lambda)\phi^b(\mathbf{x}_i)'\phi^b(\mathbf{x}_i) \\ = \; & max_i \lambda\mathcal{K}_a(\mathbf{x}_i, \mathbf{x}_i) + (1-\lambda)\mathcal{K}_b(\mathbf{x}_i, \mathbf{x}_i) \\ \leq \; & \lambda R_a^2 + (1-\lambda)R_b^2 \\ := \; & R(\lambda)^2\end{aligned}$$

Finally, observe that the slack variables of the resulting classification are given by

$$\begin{aligned}\xi_i(\lambda, \Gamma) &= \left[1 - y_i(\mathbf{w}(\lambda, \Gamma)'\phi_\lambda(\mathbf{x}_i) + \lambda b^a + (1-\lambda)b^b)\right]_0 \\ &= \Delta_i(\lambda, \Gamma)\end{aligned}$$

The result follows. ∎

To give some intuition about the dynamics of the bound, consider the following example. The training set consists of 6 examples. For both kernels $\mathcal{K}_a$ and $\mathcal{K}_b$ the SVM with $C = 1$ achieves $\delta_a^2 = \delta_b^2 = 1/18$, $R_a^2 = R_b^2 = 1$, and $\sum\xi_i^a = \sum\xi_i^b = 2$. Using a combination of the two kernels with $\lambda = 0.5$, the following table show the margins $\gamma_{a,i}$, $\gamma_{b,i}$, and the combined margin $\gamma_i = 0.5\gamma_{a,i} + 0.5\gamma_{b,i}$ for each of the 6 examples.

i	$\gamma_{a,i}$	$\gamma_{b,i}$	γ_i	$\Delta_i(0.5, 1.0)$	$\Delta_i(0.5, 1.5)$
1	1.0	2.0	1.5	0.0	0.0
2	2.0	0.0	1.0	0.0	0.5
3	0.0	1.0	0.5	0.5	1.0
4	1.0	1.0	1.0	0.0	0.5
5	3.0	0.0	1.5	0.0	0.0
6	0.0	3.0	1.5	0.0	0.0

The table also shows the SV Redundancy at $\Gamma = 1.0$ and $\Gamma = 1.5$. When a support vector (i.e. margin equal to 1) in one representation meets a non support vector in the other representation (i.e. margin greater than 1), their combined margin is larger than 1. This leads to low SV Redundancy even for large values of Γ, in particular when the margin of the non support vector is large (e.g. examples 5 and 6). A similar effect occurs also when training errors meet non support vectors. Overall, with $\Gamma = 1.5$ and $C' = 1$ the theorem bounds the value of B for the combined classifier with $18/1.5^2 + 2 \cdot 2 = 12$. Note that the value of B for the original classifiers was 20.

While the bound in the theorem is not necessarily tight in practice, it shows two major influence factors for soft-margin reduction when combining kernels. First, the soft margin decreases the lower the SV redundancy on the training set. Note that the SV redundancy on an example $\mathbf{x}_i$ decreases for some $\Gamma \geq 1$, if $\mathbf{x}_i$ is a Support Vector for one of the kernels, but not for the other. Second, the soft margin increases compared to using kernel $\mathcal{K}_a$ by itself the larger R_b^2/δ_b^2 compared to R_a^2/δ_a^2, and vice versa.

To get to a simple rule for combining kernels that can easily be applied in practice, our intuitive conclusion from the theoretical results is as follows. Combining kernels in an SVM is beneficial, if both kernels achieve approximately the same performance individually (e.g. $R_b^2/\delta_b^2 \approx R_b^2/\delta_b^2$ with similar training error) while their Support Vectors are different. In the following, we will evaluate experimentally in how far this prediction of the theory applies in practice. Our hope is that this rule does capture the major influence factors, making it an easy to use heuristic for guiding the construction of kernels in practice. The rule would already be valuable, if it could eliminate highly suboptimal design

	individual			combined ($\lambda = 0.5$)	
error (%)	BOW$_{bin}$	BOW_{tfidf}	BOL	BOW$_{bin}$/BOL	BOW_{tfidf}/BOL
course	4.33 ± 0.07	5.89 ± 0.05	4.33 ± 0.12	2.75 ± 0.09	4.40 ± 0.18
student	9.57 ± 0.16	13.55 ± 0.11	1.10 ± 0.03	1.28 ± 0.04	1.65 ± 0.05
faculty	2.67 ± 0.10	3.79 ± 0.06	2.12 ± 0.09	1.60 ± 0.06	2.57 ± 0.08

Table 1. Error rates of the word, the link, and the combined kernels on 800 training examples.

	individual			overlap	
#SV	BOW$_{bin}$	BOW_{tfidf}	BOL	BOW$_{bin}$/BOL	BOW_{tfidf}/BOL
course	158.0	413.4	519.3	116.3 (102.6)	285.4 (268.3)
student	273.6	618.1	458.9	158.3 (156.9)	359.6 (354.6)
faculty	127.7	368.3	290.1	61.3 (46.3)	146.8 (133.6)

Table 2. Number of support vectors for the word and the link kernels on 800 training examples, as well as the number of common support vectors. The number is parenthesis is the expected overlap, if the SV sets were independent.

choices so that formal and more expensive model selection (e.g. cross-validation) could be focused on the most promising design options.

4. Experiments

The experiments in this section analyse the properties of combined kernels on a Web-page classification task and evaluate how the results relate to the theory.

Experimental Setup The experiments are based on the WebKB dataset collected at CMU. It consists of the pages from the computer science departments of four universities (Cornell, University of Washington, University of Texas, and University of Wisconsin). The particular setup used here was compiled by Sean Slattery[1] for the experiments in (Slattery & Mitchell, 2000). It consists of three binary classification tasks, namely identifying course, faculty, and student home pages. There is a total of 4186 examples.

The pages in the corpus contain approximately 41.000 different words and 10.000 links within the universities. We will use two different BOW representations in the following. The corpus comes with a bag-of-words representation including 623 terms with binary word weighting (BOW$_{bin}$). Their selection is described by Slattery and Mitchell (2000). In addition, we will use a "tfc" TFIDF-weighted representation (Salton & Buckley, 1988)) using all words (BOW$_{tfidf}$). The hyperlink-based representation (BOL) is generated from incoming hyperlinks as described in Section 2.2. For both the word and the hyperlink representations, each feature vector is normalised to unit length.

[1] http://www.cs.cmu.edu/~WebKB/ICML2000-data.html

The results presented in the following are averages over 10 random test/training splits of the full 4186 document corpus. While the evaluation is done in terms of error rate here, the results are qualitatively the same also for the F_1-measure (see e.g. Joachims (2001) for a definition of F_1). The value of the SVM parameter C was chosen by optimising leave-one-out error on the training set for values $C \in \{0.01, 0.1, 1.0, 10, 100\}$. The selection of C by leave-one-out was performed for each individual test/training split using the algorithm described by Joachims (2001). The error rates reported in the following are estimated from the test set.

Can the Combination of Kernels Improve Accuracy? Table 1 shows the average error rates for the two bag-of-words, the hyperlink, and the combined kernels with $\lambda = 0.5$. Comparing the two individual bag-of-words kernels BOW$_{bin}$ and BOW$_{tfidf}$, the kernel with binary weighting exhibits lower error rates on all three tasks. This is consistent with the findings of Joachims (2001) on a similar Web-page classification task. The link kernel BOL shows approximately the same performance as BOW$_{bin}$ on "course" and "faculty", while being substantially better on "student".

The combined kernel BOW$_{bin}$/BOL outperforms each individual kernel on the tasks "course" and "faculty", while not improving performance for "student". For the "student" task the bag-of-words representation performs particularly poorly. This is in contrast to the hyperlink representation, which is particularly well-suited. The combined BOW$_{tfidf}$/BOL kernel does not lead to improved performance on any of the three tasks. However, the loss of performance for the combined kernel is small and far below the arithmetic average of the error rates of the individual kernels.

error (%)	individual BOW$_{bin}$	individual RBF	combined BOW$_{bin}$/RBF
course	4.33 ± 0.07	4.15 ± 0.09	4.10 ± 0.06
student	9.57 ± 0.16	9.26 ± 0.12	9.84 ± 0.13
faculty	2.67 ± 0.10	3.20 ± 0.07	2.75 ± 0.04

(a)

#SV	individual BOW$_{bin}$	individual RBF	overlap BOW$_{bin}$/RBF
course	158.0	285.5	157.6 (56.3)
student	273.6	427.3	273.3 (146.1)
faculty	127.7	245.4	127.5 (39.2)

(b)

Table 3. (a) Error rate for the BOW$_{bin}$, the RBF, and the combined kernel. (b) Numbers of support vectors and their overlap. All results are for $\lambda = 0.5$ and 800 training examples.

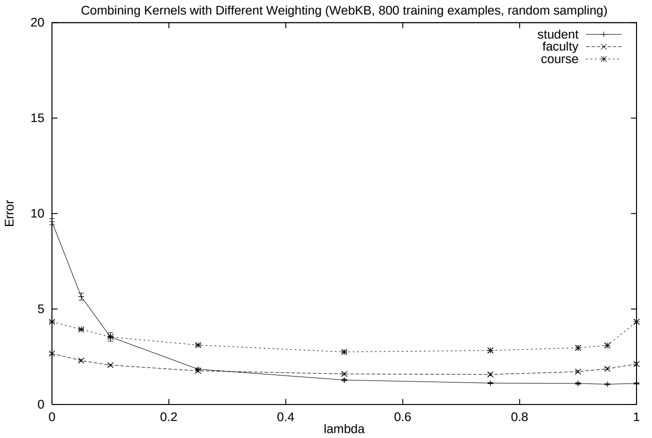

Figure 1. Error rate depending on the value of λ for the BOW$_{bin}$/BOL-combined kernel on 800 training examples.

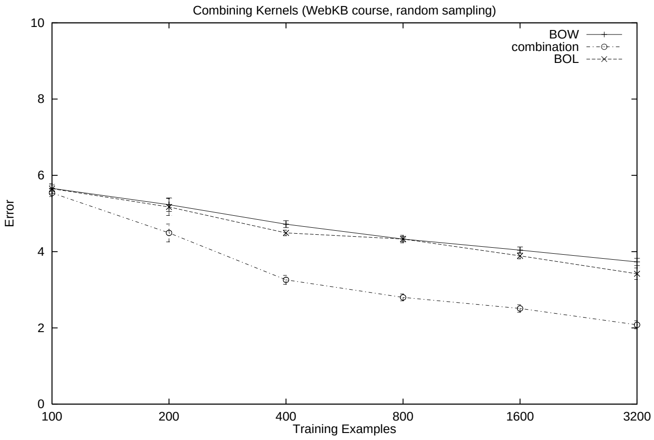

Figure 2. Error rate depending on the size of the training set for "course" using BOW$_{bin}$, BOL, and $\lambda = 0.5$.

How do the Experimental Results Relate to the Theoretical Results? The theoretical results indicate that the performance of the individual classifiers and their SV overlap are indicators for the performance of the combined kernel. In particular, the combined classifier will improve performance compared to each individual kernel, if both individual kernels have approximately the same error rate, and if their SV-overlap is low.

The experimental results are consistent with this prediction. The combined kernel performs relatively best if both individual kernels have a similar error rate, and when the SV overlap is small. Table 2 shows the number of support vectors for each individual kernel, as well as the number of support vectors that the BOL and the BOW kernels share. The overlap is small compared to the number of SVs of the individual kernels. For both combined kernels the overlap is substantially less than each individual number of SVs, and it is only slightly higher than the expected overlap assuming that the SVs were drawn at random (in parenthesis).

To construct an example with high SV overlap consider combining the BOW$_{bin}$ kernel with an RBF-Kernel (see (Cristianini & Shawe-Taylor, 2000)) with $\sigma = 0.5$ on the same representation. The error rates and the SV overlap are given in Table 3. While for these two kernels the individual error rates are very similar, there is almost perfect SV overlap between the RBF and the BOW kernel. As predicted by the theory, the combined kernel does not show substantial improvements.

How Important is the Relative Weight λ? In the previous experiments, kernels were combined with equal relative weight. Figure 1 shows the error rates dependent on the value of λ for BOW$_{bin}$/BOL. While there was no improvement for "student" in combining kernels with $\lambda = 0.5$, the plot indicates a very slight improvement for $\lambda = 0.95$. The respective plot for BOW$_{tfidf}$/BOL reveals a similarly small gain "course" for $0.9 \leq \lambda \leq 0.95$, but no improvement for the other two tasks. Similarly, combining BOW$_{bin}$/RBF the respective plot does not show substantial improvements for any value of λ.

Overall, the plot shows a surprisingly low dependency on λ over much of its range. Either combining helps for almost any value of λ, or no value of λ leads to a substantial improvement.

Does the Improvement Depend on the Number of Training Examples? The previous results were for training sets of 800 examples. As a typical example, Figure 2 shows the learning curve for "course" using BOW_{bin}, BOL, and their combination. For small training sets combining kernels does not improve performance. With increasing training set size the improvement first grows and then saturates. In terms of SV overlap this behaviour can be explained as follows. For small training set, the fraction of training examples that become SVs is high. On training sets of size 100, the BOL kernel leads to 85 and the BOW_{bin} kernel to 45 support vectors on average. The number of shared SVs is 41, leading to a similar amount of overlap as for the RBF kernel. The amount of overlap then decreases with an increasing number of training examples (see e.g. Table 2).

5. Conclusions

We have introduced a new type of kernel whose Gram matrix is the co-citation matrix known in bibliometrics. We have shown how such matrix can be combined with the document-by-document similarity matrix known in information-retrieval, to obtain a valid kernel that can perform better that the other two considered separately. We have then addressed the general issue of combining kernels, obtaining some general conditions in which a combined kernel can be expected to generalise better than a single one.

This technique is very promising in fields in which there is no real understanding of a good similarity measure between data, but several independent approaches seem to perform well (e.g. in image retrieval one can retrieve by color, texture, shapes ...). If one has many independent kernels that show equal performance, it is possible to combine them in a way to obtain a stronger one. In addition to evaluating combined kernels also in other domains, theoretical aspects need to be further pursued to get more refined conditions. Further work can be found in (Cristianini et al., 2001).

References

Ali, K., & Pazzani, M. (1996). Error reduction through learning multiple descriptions. *Machine Learning*, *24*.

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. *Annual Conference on Computational Learning Theory (COLT-98)* (pp. 92–100).

Chakrabarti, S., Dom, B., & Indyk, P. (1998). Enhanced hypertext categorization using hyperlinks. *ACM SIGMOD.*

Chang, H., Cohn, D., & McCallum, A. (2000). Creating customized authority lists. *International Conference on Machine Learning.*

Cohn, D., & Hofmann, T. (2000). The missing link – a probabilistic model of document content and hypertext connectivity. *Neural Information Processing Systems (NIPS 00).*

Cortes, C., & Vapnik, V. N. (1995). Support–vector networks. *Machine Learning Journal*, *20*, 273–297.

Craven, M., Slattery, S., & Nigam, K. (1998). First-order learning for web-mining. *European Conference on Machine Learning.*

Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods.* Cambridge University Press.

Cristianini, N., Shawe-Taylor, J., & Elisseeff, A. (2001). *On optimizing kernel alignment* (Technical Report NC2-TR-2001-087). Neurocolt.

Joachims, T. (2001). *The maximum-margin approach to learning text classifiers: Methods, theory, and algorithms.* Doctoral dissertation, Universität Dortmund.

Kleinberg, J. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM*, *46*.

Ng, K. B., & Kantor, P. (1998). An investigation of the conditions for effective data fusion in information retrieval: A pilot study. *61th Annual Meeting of the American Society for Information Science.*

Page, L., & Brin, S. (1998). Pagerank, an eigenvector based ranking approach for hypertext. *21st Annual ACM/SIGIR International Conference on Research and Development in Information Retrieval.*

Saitoh, S. (1988). *Theory of reproducing kernels and its applications.* Longman Scientific & Technical.

Salton, G., & Buckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing and Management*, *24*, 513–523.

Shawe-Taylor, J., & Cristianini, N. (2000). Margin distribution and soft margin. In *Advances in large margin classifiers*, 349–358. Cambridge, MA: MIT Press.

Slattery, S., & Mitchell, T. (2000). Discovering test set regularities in relational domains. *International Conference on Machine Learning.*