Document Classification Using Multiple Views

Albert Gordo
Computer Vision Center
Universitat Autònoma de Barcelona, Spain
agordo@cvc.uab.es

Florent Perronnin
Xerox Research Centre Europe, France
florent.perronnin@xrce.xerox.com

Ernest Valveny
Computer Vision Center
Universitat Autònoma de Barcelona, Spain
ernest.valveny@cvc.uab.es

***Abstract*—The combination of multiple features or views when representing documents or other kinds of objects usually leads to improved results in classification (and retrieval) tasks. Most systems assume that those views will be available both at training and test time. However, some views may be too 'expensive' to be available at test time. In this paper, we consider the use of Canonical Correlation Analysis to leverage 'expensive' views that are available only at training time. Experimental results show that this information may significantly improve the results in a classification task.**

***Keywords*-Document classification; multiple views; run-lengths; CCA;**

I. Introduction

The combination of multiple features to represent documents or other kinds of objects such as natural images to improve classification results is a common technique widely used in the literature [16, 12, 1, 14]. For example, when dealing with documents, these features could be textual (such as a bag-of-words computed with the output of an OCR [6]), layout-based (such as a tree or a graph [13]), visual (such as features based on texture analysis [5]), or low-level features obtained, for example, after a connected components analysis, such as average weight, height, or density in different regions of the document [16]. In the following, we will refer to any representation that can be extracted from an object as a view of the object. Many different views can typically be extracted from the same object.

Several strategies exist to combine these views, ranging from concatenating all the views (assuming they all can be represented as feature vectors) and training a classifier in the combined space, to more advanced frameworks such as Multiple Kernel Learning [11].

What these strategies have in common is that they assume that all the views of an object can be obtained both at training and testing stages. However, this is not always possible since some views may be 'expensive' and we cannot afford to obtain them at test time for every object.

If we take the example of the textual features, running an OCR on a single page may take up to a few seconds per page when aiming at a high quality output, depending on the quality of the input document, slowing down the whole document distribution workflow. This OCR system could also work with a per-page fee license, and so obtaining a view that relies on OCR will be monetarily expensive. In a completely different scenario, obtaining a view in medical imaging may require an invasive procedure that is discouraged.

We will refer to those views we cannot usually afford at test time as *costly* views, and the ones we can afford at test time as *cheap* views. Fortunately, in some scenarios, we are able to collect, *off-line*, both the cheap and costly views of some objects, and this can be exploited to train more discriminative classifiers.

Some works exist about leveraging information not available at test time. In the recent [10], the authors explore the use of weakly-paired multimodal data (*i.e.*, the views may have different numbers of items and not be perfectly aligned) in a dimensionality reduction and transfer learning context. Image and audio features are available at training time, and the goal is to perform dimensionality reduction of the audio features, which are the only ones available at test time. In [8], Canonical Correlation Analysis (CCA) is used to retrieve images based on text queries, where the image features are no longer available at test time. In [2], a kernelized CCA is also used to improve the clustering of images and text.

In this paper, our goal is to exploit these costly views, that are available for some documents only at training time, to train a more discriminative classifier that deals only with cheap views at test time. We will use Canonical Correlation Analysis to achieve this purpose. Although CCA has been used in similar scenarios for retrieval [8] and clustering [2] as we just mentioned, we are not aware of it being used in classification tasks.

The main idea behind the method is to find a common subspace between cheap and costly views and a set of projection vectors to embed the views into that subspace. After that, the cheap views can be projected into that common subspace and be used to train a classifier. At test time, we will project the cheap views into the subspace and use the classifier without any need to access the costly view. Since we used the costly views to find this subspace, the projected cheap views are more discriminative than the original ones.

The rest of the paper is organized as follows. Section II overviews the CCA principles, as well as its limitations and extensions. Section III shows how to use CCA to train a classifier that works in the cheap views domain but exploits the costly views information available at training time. Section IV deals with the experimental evaluation, and finally Section V concludes the paper.

II. Canonical Correlation Analysis

Canonical Correlation Analysis (CCA) is a well-known tool for the analysis of multi-view data, which was first introduced by Harold Hotelling in 1936 [9]. CCA and its variants have been used, for example, in unsupervised tasks such as retrieval [8], clustering [2], or supervised dimensionality reduction [7].

Let us first consider a set of N unlabeled training samples, and let $A \in \mathbb{R}^{d_1 \times N}$ and $B \in \mathbb{R}^{d_2 \times N}$ be two views of the data represented with column feature vectors. The dimensionality of the vectors in A and B may be different. Without loss of generality, we will assume that A is the *cheap* view and that B is the *expensive* view of the samples. Let us also define the covariance matrices $C_{aa} = AA'$, $C_{bb} = BB'$, $C_{ab} = AB'$, and $C_{ba} = C'_{ab}$.

The goal of CCA is to find a projection of each view that maximizes the correlation between the projected representations. This can be expressed as:

$$\operatorname*{argmax}_{a,b} \frac{a'AB'b}{\sqrt{a'AA'a}\sqrt{b'BB'b}} = \frac{a'C_{ab}b}{\sqrt{a'C_{aa}a}\sqrt{b'C_{bb}b}}, \qquad (1)$$

which can be rewritten as

$$\operatorname*{argmax}_{a,b} a'C_{ab}b, \qquad (2)$$

subject to the constraints $a'C_{aa}a = 1$ and $b'C_{bb}b = 1$.

Here, $a \in \mathbb{R}^{d_1}$ and $b \in \mathbb{R}^{d_2}$ are the projections that embed the data from A and B into a one-dimensional common subspace where the correlation is maximal. Usually we will be interested in a subspace of k dimensions instead of only one. To do so, we will need to solve Equation (1) k times to obtain the projection vectors $\{a_1, a_2, \ldots, a_k\}$ and $\{b_1, b_2, \ldots, b_k\}$, subject to them being uncorrelated.

To solve Equation (1), we first rewrite it modeling the constraints with Lagrangian multipliers. Then we take partial derivatives with respect to a and b, set them equal to 0 and solve the system [8]. Solving for a leads to the following symmetric eigenvalue problem:

$$Za_k = \lambda_k^2 a_k, \qquad (3)$$

with $Z = C_{aa}^{-1} C_{ab} C_{bb}^{-1} C_{ba}$.

Solving Equation (3) gives the $\{a_1, a_2, \ldots a_k\}$ projection vectors that project the cheap view A into the k-dimensional common subspace. Similarly, we can solve for b and arrive to an equation analogous to (3) to obtain the $\{b_1, b_2, \ldots b_k\}$ projection vectors that project the expensive view B into the k-dimensional common subspace. Note that, since we will only project the cheap view into the common subspace (as the expensive view will not be available at test time), we only need to solve for the a's.

A. Limitations and extensions of the CCA

As presented, CCA suffers from two important limitations. First, it is restricted to only two views, and, second, it is restricted to vectorial data. Both limitations can be lifted using common CCA extensions.

To solve CCA using multiple views, the cross-covariance matrices can be combined to form an extended matrix. Solving a generalized eigenvalue problem that makes use of this matrix yields the projection vectors of each view [2].

When data is not in vectorial form (such as documents represented with a layout graph, or descriptors based on variable-length sequences), or is not linearly separable, a kernelized CCA (KCCA) can be used in the kernel space [8]. This follows a formulation very similar to the original CCA, although a regularization factor must be added to avoid trivial solutions [8].

III. Learning with CCA

In this section we will first review the process of training and classifying when only two views are available, one cheap and one costly. We will assume that we have access to a set of objects where both views U_{cheap} and U_{costly} are available. Note that the labels of such documents are not needed. We also have access to the cheap view of a set of objects, S_{cheap}, where the labels l are available.

The training process is explained in Algorithm 1. First, the common subspace between the cheap and costly views is learned using the unsupervised data. This produces a set of projection vectors with which the supervised data S_{cheap} is projected into the common subspace. Then, a classifier (for example an SVM) is learned in this space.

The classification process is explained in Algorithm 2. A new unlabeled sample x_{cheap} is first projected into the common subspace using the learned projections, and then classified using the trained classifier.

In the case of non-vectorial representations, CCA can be easily replaced with KCCA in Algorithm 1. This can be trivially extended to more views. For example, assuming we have m cheap views and n costly views, at *train* time, CCA in Algorithm 1 will return $m+n$ projections, one for each view. We will project only the m cheap views with their corresponding projection and train one independent classifier for each of the projected cheap views, m in total.

At *test* time, we will project each of the m cheap views with their projection and classify each one with their classifier. The final score can be computed, for example, averaging the scores of the m classifiers.

IV. Experiments

A. Experimental setup

Unfortunately, we are not aware of any public documents dataset where multiple views could be easily exploited.

Algorithm 1 Train classifier

```
Input: $U_{cheap} \in \mathbb{R}^{d_1 \times N}$, cheap view of the unsupervised data,
    $U_{costly} \in \mathbb{R}^{d_2 \times N}$, costly view of the unsupervised data,
    $k$, s.t. $k \leq min(d_1, d_2)$, dimensionality of the embedded space
    $S_{cheap} \in \mathbb{R}^{d_1 \times M}$, cheap view of the supervised data, and
    $L \in \mathbb{R}^{M}$, labels of the supervised data.
Output: $a \in \mathbb{R}^{k \times d_1}$, the projection matrix that embeds S into the common subspace of $k$ dimensions, and
    $W \in \mathbb{R}^{k}$, the trained classifier in the embedded space.

    1- Obtain the projections of the cheap view into the common subspace with CCA. The projections of the costly view, $b$, can be discarded:
    $[a,b] = CCA(U_{cheap}, U_{costly}, k)$
    2- Project the supervised data into the subspace:
    $P = a \cdot S_{cheap}$
    3- Train a classifier in the embedded space:
    $W = TrainClassifier(P, L)$
```

Algorithm 2 Classify sample

```
Input: $x_{cheap} \in \mathbb{R}^{d_1}$, cheap view of an unlabeled sample to classify,
    $a \in \mathbb{R}^{k \times d_1}$, the projection matrix that embeds $s_{cheap}$ into the common subspace of $k$ dimensions, and
    $W \in \mathbb{R}^{k}$, the trained classifier in the embedded space.
Output: $l$, the label of the input document $x$.

    1- Project $x_{cheap}$ into the embedded subspace:
    $P = a \cdot x_{cheap}$
    2- Classify the sample:
    $l = Classify(P, W)$
```

Therefore, all our experiments have been carried out in an in-house dataset. This in-house dataset comes from real-world data, and contains approximately 40,000 document images split into 181 categories, mostly letters and forms. The number of documents in each category varies significantly, from as few as 5 documents in one category to as many as 4,000 in another.

For each document, two views are available. First, the cheap view, a multi-scale runlength histogram of 1,512 dimensions. This histogram captures the visual appearance of the page at several positions and scales, providing some basic structural information. Second, the costly view, a bag-of-words histogram of 5,000 dimensions constructed with the text output of an OCR application. The OCR bag-of-words histogram is the costly view both because of the economic costs associated with the licensing of a third party software, and because it takes up to a few seconds per page to obtain the descriptor. The runlength histograms, however, can be computed in a few tens of milliseconds.

Both histograms are L_1 normalized and then square rooted. It has been shown that square rooting the histograms when using a linear kernel corresponds to an explicit embedding of the Bhattacharyya similarity [15], significantly improving the results at virtually no cost.

As we will see in the experiments, the textual bag-of-words has a significantly better accuracy than the visual features. However, the computational cost and a per-page fee makes its use for all documents inviable at test time.

We divide the documents in 3 different sets:

Test set: The test set contains approximately 100 documents of each class. For some categories this is not possible since they contain less documents, but we ensure that it contains at least one document of each category. Only the cheap view (runlengths) is available.

Supervised training set: We vary the size of the supervised training set, from 5 to 20 documents per class. Only the cheap view is available, but we have access to the true labels of the documents.

Unsupervised training set: The rest of the documents, combined with the supervised training set, compose the unsupervised training set. Both cheap and costly views are available, but the labels of the documents are not available.

We repeat the experiments with 5 different test/supervised training/unsupervised training partitions and average the results. For computing the CCA projections, we used the code available at [3]. For the supervised classification, we used a linear SVM trained with Stochastic Gradient Descent [4]. Since we square rooted the vectors, this is equivalent to using a SVM with a Bhattacharyya kernel.

We perform two different experiments. The first one is standard classification without rejection, where we report the mean class accuracy, *i.e.*, computing the accuracy of each category independently and then averaging the results. When dealing with very unbalanced categories like in this dataset, the mean class accuracy gives a more meaningful result than the mean document accuracy.

In the second experiment we introduce rejection: we only assign a label to a document if the classification score is higher than a threshold, and otherwise we reject the document. This can ensure a very high classification rate, although it may lead to the rejection of many documents. To represent these results we use accuracy-coverage plots, reminiscent of the precision-recall plots used in retrieval. These plots show what is the expected coverage (*i.e.*, percentage of documents that we label, correctly or not) when aiming at a given average classification rate.

B. Classification without rejection

Results using 5, 10, and 20 supervised training samples per class can be seen in Figures 1-3. We plot the classification results using only the cheap, visual features, as well as the results after embedding the samples into the

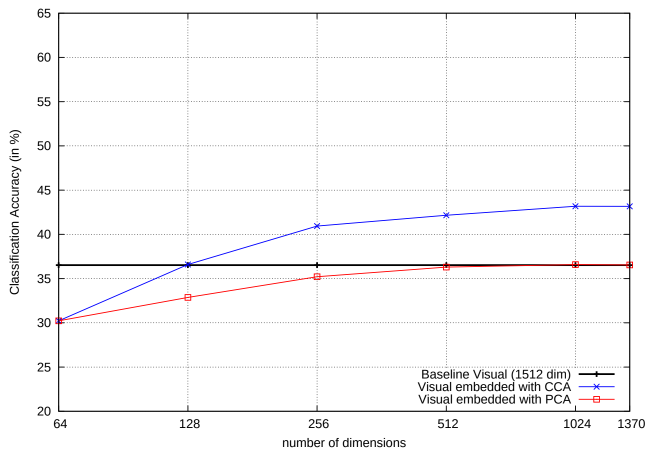


Figure 1. Classification results using 5 training samples per class. The text baseline (which cannot be computed in practice) is 73.07%

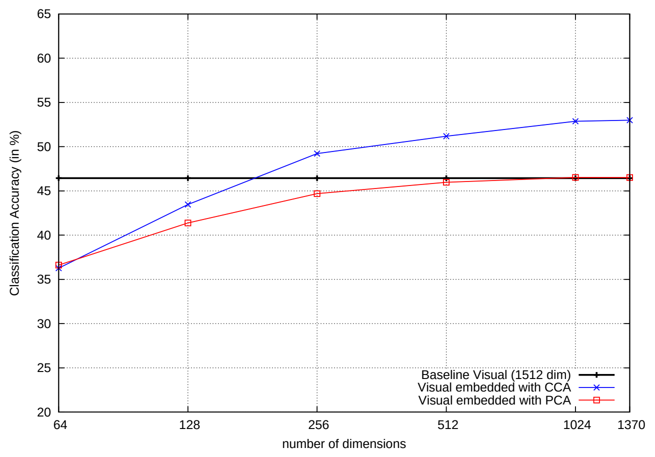


Figure 2. Classification results using 10 training samples per class. The text baseline (which cannot be computed in practice) is 79.21%

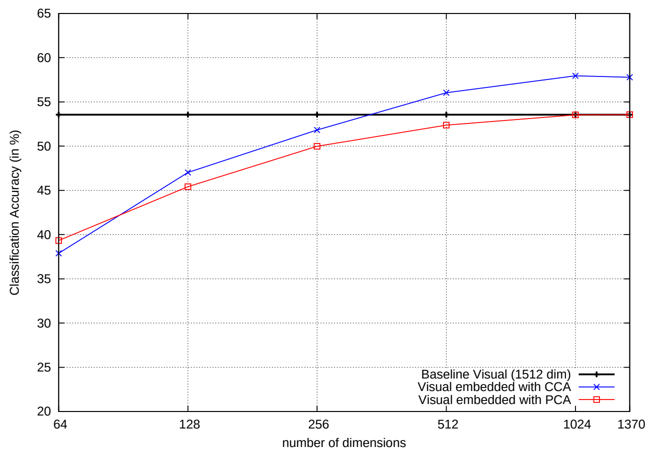


Figure 3. Classification results using 20 training samples per class. The text baseline (which cannot be computed in practice) is 82.51%

common subspace with CCA, as a function of the number of dimensions of the subspace. The maximum number of dimensions is limited by the minimum number of independent dimensions in the original views, in this case 1,370. This is due to the fact that the runlength histograms contain a significant amount of zeros, limiting the number of independent dimensions. Since we are also performing dimensionality reduction, we also plot the classification results obtained using PCA instead of CCA. In this case, the expensive view is not used. Finally, we also report the results using the text features. Note that these are the expensive features, and would not be available at test time. This corresponds to a hypothetical upper bound of the system. We can draw the following conclusions:

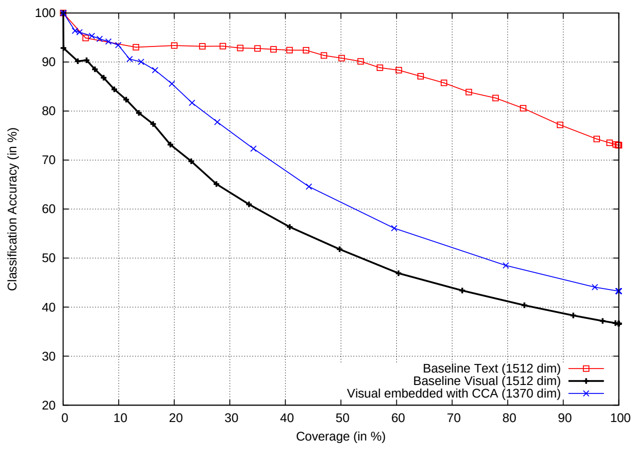
Cheap *vs.* costly baselines: As expected, the costly features perform significantly better than the cheap features. This is essentially by design: a costly feature that performs worse than a cheap one would probably not be considered in the first place.

Visual *vs.* visual after CCA embedding: We can observe how, in all three settings, using CCA can noticeably improve the baseline results when using the maximum number of dimensions available in the subspace. When performing dimensionality reduction, we can significantly reduce the dimensionality while still obtaining better results than the baseline. This is particularly true in the case when few supervised samples are available: with 5 samples per class, we can reduce the descriptors down to 128 dimensions and still obtain the same results as the baseline. However, when using 20 documents per class for training, we can only reduce down to 512 dimensions. This suggests that this CCA embedding is particularly suited in the case where little supervised data is available.

CCA *vs.* PCA embeddings: In all three settings, using CCA for dimensionality reduction generally produces better results than using PCA. The differences are very small when reducing to a very low number of dimensions (and, in fact, PCA performs slightly better than CCA at 64 dimensions with 20 training samples per class), but increasing the number of dimensions also increases the differences between PCA and CCA. This is not unexpected, since one of the main uses of CCA is precisely to perform a dimensionality reduction. Note how, as in the previous case, the differences also become smaller when using a larger number of supervised documents for the learning stage, supporting the idea that CCA is particularly suited when few supervised samples are available.

C. Classification with rejection

In practice, the classification results that we have observed are not useful in a real system. Usually, rejection is integrated in the system: if the score of a sample does not reach a given threshold, the sample will not be classified in this stage, and will be sent to a different pipeline, which will probably use more expensive features or human intervention. Typically, to guarantee a high accuracy, we set a high



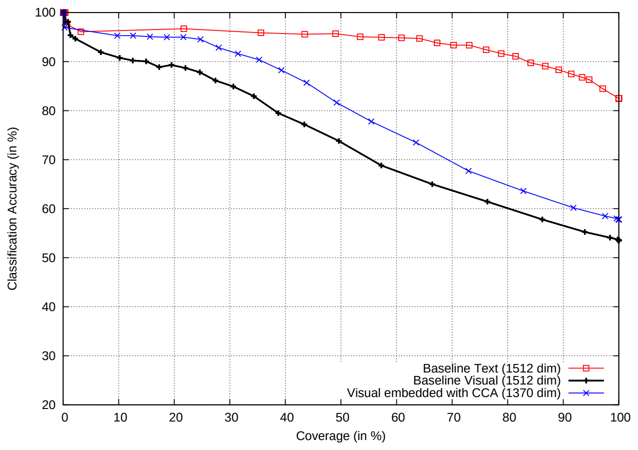


Figure 4. Accuracy-coverage plot – Top: 5 labeled training samples per class. Bottom: 20 labeled training samples per class.

threshold, but then most of the documents will be rejected, leading to a low coverage. On the other hand, setting a low score will yield a high coverage, but the classification rate will drop.

Figure 4 show accuracy-coverage plots for the visual baseline as well as text baseline and the visual features embedded in a 1,370 dimensional subspace with CCA, using 5 and 20 training samples per class. We can observe that:

i) Given an accuracy threshold, the coverage with CCA is significantly larger than the visual baseline. Aiming at a 90% classification accuracy and using 5 training samples per class, we can cover approximately 10% more of the dataset when using CCA (4% to 14%). When using 20 training samples per class, we can cover approximately 20% more of the dataset (from 15% to 35%). Since all the rejected documents will be sent to the more expensive pipeline with textual features, these quantities can be directly translated into savings.

ii) The visual CCA and the textual baseline perform very similarly when the goal is to obtain a very high precision accuracy. This is particularly relevant because it shows that, depending on the objectives, the expensive view can be replaced with the embedded cheap view without significant loss.

V. Conclusions

In this paper we have shown how Canonical Correlation Analysis (CCA) can be used to improve the accuracy in a document classification task where some views of the data are only available at the training stage, since they are too ‘expensive’ to be obtained at test time. Finally, we have shown how, in a system with rejection, CCA can significantly increase the coverage over the baseline when aiming at the same classification accuracy.

Acknowledgment

Albert Gordo and Ernest Valveny are partially supported by the Spanish projects TIN2009-14633-C03-03, and CONSOLIDER-INGENIO 2010 (CSD2007-00018).

References

[1] A. Behera, D. Lalanne, and R. Ingold. Combining color and layout features for the identification of low-resolution documents. *IJSP*, 2005.

[2] M. B. Blaschko and C. H. Lampert. Correlational spectral clustering. In *CVPR*, 2008.

[3] M. Borga. Canonical correlation: a tutorial, 2001. http://www.imt.liu.se/~magnus/cca/.

[4] L. Bottou. SGD. http://leon.bottou.org/projects/sgd.

[5] J. F. Cullen, J. J. Hull, and P. E. Hart. Document image database retrieval and browsing using texture analysis. In *ICDAR*, 1997.

[6] D. Doermann. The indexing and retrieval of document images: A survey. *CVIU*, 1998.

[7] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *CVPR*, 2011.

[8] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis; an overview with application to learning methods. Technical report, Department of Computer Science, Royal Holloway, University of London, 2003.

[9] H. Hotelling. Relations between two sets of variates. *Biometrika*, 1936.

[10] C. Lampert and O. Kromer. Weakly-paired maximum covariance analysis for multimodal dimensionality reduction and transfer learning. In *ECCV*, 2010.

[11] G. Lanckriet, N. Cristianini, L. E. Ghaoui, P. Bartlett, and M. Jordan. Learning the kernel matrix with semi-definite programming. *JMLR*, 2004.

[12] L. Likforman-Sulem, P. Vaillant, and F. Yvon. Proper names extraction from fax images combining textual and image features. In *ICDAR*, 2003.

[13] S. Marinai, E. Marino, and G. Soda. Layout based document image retrieval by means of xy tree reduction. In *ICDAR*, 2005.

[14] M.-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *CVPR*, 2006.

[15] F. Perronnin, J. Sánchez, and Y. Liu. Large-scale image categorization with explicit data embedding. In *CVPR*, 2010.

[16] C. Shin, D. Doermann, and A. Rosenfeld. Classification of document pages using structure-based features. *IJDAR*, 2001.