Names and Faces in the News

Tamara L. Berg, Alexander C. Berg, Jaety Edwards, Michael Maire,
Ryan White, Yee-Whye Teh, Erik Learned-Miller and D.A. Forsyth

Computer Science Division
U.C. Berkeley
Berkeley, CA 94720
daf@cs.berkeley.edu

Abstract

We show quite good face clustering is possible for a dataset of inaccurately and ambiguously labelled face images. Our dataset is 44,773 face images, obtained by applying a face finder to approximately half a million captioned news images. This dataset is more realistic than usual face recognition datasets, because it contains faces captured "in the wild" in a variety of configurations with respect to the camera, taking a variety of expressions, and under illumination of widely varying color. Each face image is associated with a set of names, automatically extracted from the associated caption. Many, but not all such sets contain the correct name.

We cluster face images in appropriate discriminant coordinates. We use a clustering procedure to break ambiguities in labelling and identify incorrectly labelled faces. A merging procedure then identifies variants of names that refer to the same individual. The resulting representation can be used to label faces in news images or to organize news pictures by individuals present.

An alternative view of our procedure is as a process that cleans up noisy supervised data. We demonstrate how to use entropy measures to evaluate such procedures.

1. Introduction

It is straightforward to obtain enormous datasets of images, with attached annotations. Examples include: collections of museum material [3]; the Corel collection of images; any video with sound or closed captioning; images collected from the web with their enclosing web pages; or captioned news images.

Exploiting partially supervised data is a widely studied theme in vision research. Image regions may usefully and fairly accurately be linked with words, even though the words are not linked to the regions in the dataset originally [7, 2]. For example, models based around templates and relations may be learned from a dataset of motorcycle images where one never specifies *where* in the image the motorcycle lies (for faces, see [14, 10]; for animals in static images and in video, see [16, 18]; for a range of objects, see [9]). In this paper, we show that faces and names can be linked in an enormous dataset, despite errors and ambiguities in proper name detection, in face detection and in correspondence.

Face recognition is well studied, and cannot be surveyed reasonably in the space available. Early face recognition is done in [19, 21] and is reviewed in [12, 6, 13]. Our problem is slightly different from face recognition, in that it is more important to identify discriminant coordinates — which can be used to distinguish between faces, even for individuals not represented in the dataset — than to classify the faces. As a result, we focus on adopting the kPCA/LDA methodology, rather than on building a multi-class classifier. Our current work is a necessary precursor to real world face recognition machinery: building large and realistic sets of labelled data for recognition. We can leverage past work by using it to determine what features might be useful for identifying similar faces.

The general approach involves using unambiguously labelled data items to estimate discriminant coordinates (section 3). We then use a version of k-means to allocate ambiguously labelled faces to one of their labels (section 4). Once this is done, we clean up the clusters by removing data items far from the mean, and re-estimate discriminant coordinates (section 4.2). Finally, we merge clusters based on facial similarities (section 4.3). We show qualitative and quantitative results in section 5.

2. Dataset

We have collected a dataset consisting of approximately half a million news pictures and captions from Yahoo News over a period of roughly two years.

Faces: Using the face detector of [15] we extract 44,773 face images (size 86x86 or larger with sufficient face detection scores and resized to 86x86 pixels). Since these pictures were taken "in the wild" rather than under fixed

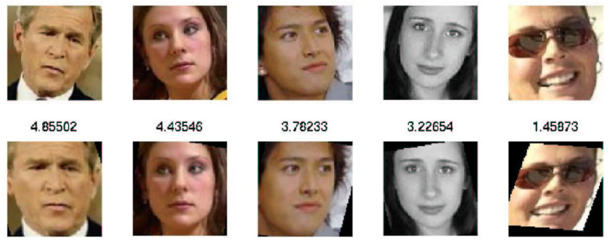


Figure 1: **Top row** *shows face detector results,* **bottom row** *shows images returned by the face rectifier with rectification score shown* **center** *(larger scores indicate better rectification performance). The face rectifier uses an SVM to detect feature points on each face. Gradient descent is then used to find the best affine transformation to map feature points to canonical locations.*

laboratory conditions, they represent a broad range of individuals, pose, expression and illumination conditions. Individuals also change over time, which has been shown to hamper recognition in [12]. Our face recognition dataset is more varied than any other to date.

Names: We extract a lexicon of proper names from all the captions by identifying two or more capitalized words followed by a present tense verb ([8]). Words are classified as verbs by first applying a list of morphological rules to present tense singular forms, and then comparing these to a database of known verbs (WordNet [25]). This lexicon is matched to each caption. Each face detected in an image is associated with every name extracted from the associated caption (e.g. fig 2). Our job is to label each face detector response with the correct name (if one exists).

Scale: We obtain 44,773 large and reliable face detector responses. We reject face images that cannot be rectified satisfactorily, leaving 34,623. Finally, we concentrate on images associated with 4 or fewer names, leaving 27,742 faces.

2.1 Distinctive Properties

Performance figures reported in the vision literature are inconsistent with experience of deployed face recognition systems (e.g., see [17]). This suggests that lab datasets lack important phenomena. Our dataset differs from typical face recognition datasets in a number of important ways.

Pose, expression and illumination vary widely. The face detector tends not to detect lateral views of faces, but (figure 3) we often encounter the same face illuminated with markedly different colored light and in a broad range of expressions. Spectacles and mustaches are common. There are wigs, images of faces on posters, differences in resolution and identikit pictures. Quite often there are multiple copies of the same picture (this is due to the way news pictures are prepared, rather than a collecting problem) or multiple pictures of the same individual in similar configurations. Finally, some individuals are tracked across time.

Name frequencies have the long tails that occur in natural language problems. We expect that face images roughly follow the same distribution. We have hundreds to thousands of images of a few individuals (e.g. *President Bush*), and a large number of individuals who appear only a few times or in only one picture. One expects real applications to have this property. For example, in airport security cameras a few people, security guards, or airline staff might be seen often, but the majority of people would appear infrequently. Studying how recognition systems perform under such a distribution is important.

The sheer **volume** of available data is extraordinary. We have sharply reduced the number of face images we deal with by using a face detector that is biased to frontal faces and by requiring that faces be large and rectify properly. Even so, we have a dataset that is comparable to, or larger than, the biggest available lab sets and is much richer in content. Computing kernel PCA and linear discriminants for a set this size requires special techniques (section 3.1).

2.2 Rectification

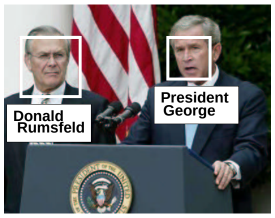
Before comparing images, we automatically rectify all faces to a canonical pose. Five support vector machines are trained as feature detectors (corners of the left and right eyes, corners of the mouth, and the tip of the nose) using 150 hand clicked faces. We use the geometric blur of [5] applied to grayscale patches as the features for our SVM. A new face is tested by running each SVM over the image with a weak prior on location for each feature. We compute an affine transformation defined by the least squares solution between maximal outputs of each SVM and canonical feature locations. We then perform gradient descent to find the affine transformation which best maps detected points to canonical feature locations. Each image is then rectified to a common pose and assigned a score based on the sum of its feature detector responses.

Larger rectification scores indicate better feature detection and therefore better rectification. We filter our dataset by removing images with poor rectification scores and are left with 34,623 face images. Each face is automatically cropped to a region surrounding the eyes, nose and mouth to eliminate effects of background on recognition. The RGB pixel values from each cropped face are concatenated into a vector and used from here on.

3. Discriminant Analysis

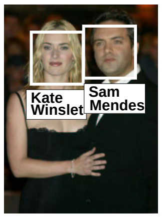
We perform kernel principal components analysis (kPCA) to reduce the dimensionality of our data and linear discriminant analysis (LDA) to project data into a space that is suited for the discrimination task.

Kernel Principal Components Analysis: Kernel PCA [20] uses a kernel function to efficiently compute a principal component basis in a high-dimensional feature space,



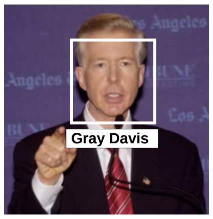
President George W. Bush makes a statement in the Rose Garden while Secretary of **Defense Donald Rumsfeld** looks on, July 23, 2003. Rumsfeld said the United States would release graphic photographs of the dead sons of **Saddam Hussein** to prove they were killed by American troops. Photo by Larry Downing/Reuters

World number one **Lleyton Hewitt** of Australia hits a return to **Nicolas Massu** of Chile at the Japan Open tennis championships in Tokyo October 3, 2002. REUTERS/Eriko Sugita

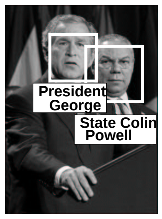


British director **Sam Mendes** and his partner actress **Kate Winslet** arrive at the London premiere of 'The Road to Perdition', September 18, 2002. The films stars **Tom Hanks** as a Chicago hit man who has a separate family life and co-stars **Paul Newman** and Jude Law. REUTERS/Dan Chung

German supermodel **Claudia Schiffer** gave birth to a baby boy by Caesarian section January 30, 2003, her spokeswoman said. The baby is the first child for both Schiffer, 32, and her husband, British film producer **Matthew Vaughn**, who was at her side for the birth. Schiffer is seen on the German television show 'Bet It...?!' ('Wetten Dass...?!') in Braunschweig, on January 26, 2002. (Alexandra Winkler/Reuters)



Incumbent California Gov. **Gray Davis** (news - web sites) leads Republican challenger **Bill Simon** by 10 percentage points – although 17 percent of voters are still undecided, according to a poll released October 22, 2002 by the Public Policy Institute of California. Davis is shown speaking to reporters after his debate with Simon in Los Angeles, on Oct. 7. (Jim Ruymen/Reuters)



US **President George** W. Bush (L) makes remarks while Secretary of **State Colin Powell** (R) listens before signing the US Leadership Against HIV /AIDS , Tuberculosis and Malaria Act of 2003 at the Department of State in Washington, DC. The five-year plan is designed to help prevent and treat AIDS, especially in more than a dozen African and Caribbean nations(AFP/Luke Frazza)

Figure 2: *Given an input image and an associated caption (images above and captions to the right of each image), our system automatically detects faces (white boxes) in the image and possible name strings (bold). We use a clustering procedure to build models of appearance for each name and then automatically label each of the detected faces with a name if one exists. These automatic labels are shown in boxes below the faces. Multiple faces may be detected and multiple names may be extracted, meaning we must determine who is who (e.g., the picture of* Cluadia Schiffer*).*

related to the input space by some nonlinear map. Kernel PCA has been shown to perform better than PCA at face recognition [23]. Kernel PCA is performed as follows: Compute a kernel matrix, K, where K_{ij} is the value of the kernel function comparing $image_i$ and $image_j$ (we use a Gaussian kernel). Center the kernel matrix in feature space (by subtracting off average row, average column and adding on average element values). Compute an eigendecomposition of K, and project onto the normalized eigenvectors of K.

Linear Discriminant Analysis: LDA has been shown to work well for face discrimination [24, 4, 12] because it uses class information to find a set of discriminants that push means of different classes away from each other.

3.1 Nyström Approximation

Our dataset is too large to do kPCA directly as the kernel matrix, K will be of size NxN, where N is the the number of images in the dataset, and involve approximately $2{*}10^9$ image comparisons. Therefore, we instead use an approximation to calculate the eigenvectors of K. Incomplete Cholesky Decomposition (ICD [1]) can be used to calculate an approximation to K with a bound on the approximation error, but involves accessing all N images for each column computation (where N is the number of images in the dataset). The Nyström approximation method (cf [22, 11]) gives a similar result, but allows the images to be accessed in a single batch rather than once for each column computation (thereby being much faster to compute for large matrices). Nyström does not give the same error bound on its approximation to K. However, because of the smoothing properties of kernel matrices we expect the number of large eigenvalues of our matrix to be small, where the number of large eigenvalues should go down relative to the amount of smoothing. In our matrix we observed that the eigenvalues do tend to drop off quickly. Because of this, a subset of the columns of our matrix, should encode much of the data in the matrix. This implies that the Nyström method may provide a good approximation to K.

The Nyström method computes two exact subsets of K, A and B, and uses these to approximate the rest of K. Using this approximation of K, the eigenvectors can be approximated efficiently.

First the N×N kernel matrix, K, is partitioned as

$$K = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}$$

with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{(N-n)\times n}$ and $C \in \mathbb{R}^{(N-n)\times(N-n)}$. Here, A is a subset of the images, (in our case 1000 ran-

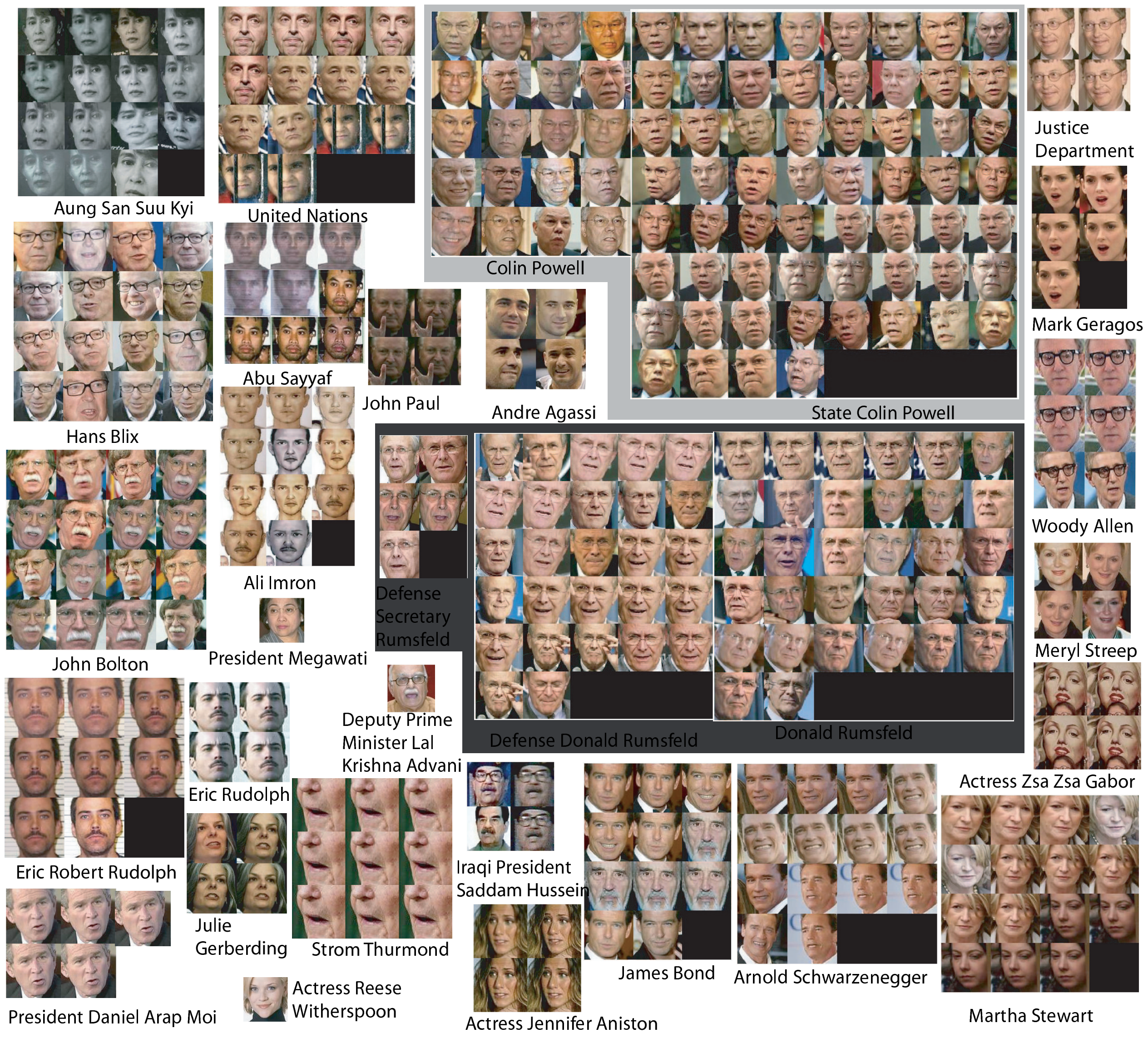


Figure 3: *The figure shows a representative set of clusters from our largest threshold series, illustrating a series of important properties of both the dataset and the method.* **Note** *that this picture greatly exaggerates our error rate in order to show interesting phenomena and all the types of error we encounter. 1: Some faces are very frequent and appear in many different expressions and poses, with a rich range of illuminations (e.g. clusters labelled* State Colin Powell*, or* Donald Rumsfeld*). These clusters also demonstrate that our clusterer can cope with these phenomena. 2: Some faces are rare, or appear in either repeated copies of one or two pictures or only slightly different pictures (e.g. cluster labelled* Abu Sayyaf *or* Actress Jennifer Aniston*). 3: Some faces are not, in fact, photographs (*Ali Imron*). 4: The association between proper names and faces is still somewhat noisy, because it remains difficult to tell which strings are names of persons (*United Nations*, which shows three separate face phenomena that co-occur frequently with this string;* Justice Department*, which is consistent as to face but not the name of a person; and* President Daniel Arap Moi *or* John Paul*, which show a names associated with the wrong face). 5: some names are genuinely ambiguous (*James Bond*, which shows two different faces naturally associated with the name (the first is an actor who played James Bond, the second an actor who was a villain in a James Bond film) . 6: Some faces appear at both low and reasonable resolution (*Saddam Hussein*). 7: Our cluster merging process is able to merge clusters depicting the same face but labelled with distinct strings (the clusters in the light gray and dark gray polygons, respectively). 8: Our cluster merging process is not perfect and could benefit from deeper syntactic knowledge of names (*Eric Rudolph *and* Eric Robert Rudolph*), but in cases such as* Defense Secretary Rumsfeld, Donald Rumsfeld *and* Defense Donald Rumsfeld *the mergings produced are correct. 9: Our clustering is quite resilient in the presence of spectacles (*Hans Blix*), perhaps wigs (*John Bolton*) and mustaches (*John Bolton*).*

Proposed Merges	
President Bush	President George
Donald Rumsfeld	Defense Secretary Donald Rumsfeld
State Colin Powell	Colin Powell
President Bush	Richard Myers
President Bush	United Nations
Defense Donald Rumsfeld	Donald Rumsfeld
Venezuelan President Hugo Chavez	Hugo Chavez

Table 1: *Multiple names can often refer to the same person. We link names to people based on images. If two names have the same associated face, then they must refer to the same person. The above pairs are the top name merges proposed by our system. Merges are proposed between two names if the clusters referring to each name contain similar looking faces.*

domly selected images) compared to themselves, B is the comparison of each of the images of A, to the rest of the images in our dataset, and C is approximated by the Nyström method. Nyström gives an approximation for C as, $\hat{C} = B^T A^{-1} B$. This gives an approximation to K, $\hat{K} = \begin{bmatrix} A & B \\ B^T & \hat{C} \end{bmatrix}$

Then we form $\tilde{K}$, the centered version of our approximation $\hat{K}$, by calculating approximate average row, average column sums (these are equal since K is symmetric), and average element values. We can approximate the average row (or column) sum as:

$$\hat{K} 1_N = \begin{bmatrix} A 1_n + B 1_{N-n} \\ B^T 1_n + B^T A^{-1} B 1_{N-n} \end{bmatrix}$$

We center as usual, $\tilde{K} = \hat{K} - \frac{1}{N} 1_N \hat{K} - \frac{1}{N} \hat{K} 1_N + \frac{1}{N^2} 1_N \hat{K} 1_N$.

We then solve for the orthogonalized approximate eigenvectors as follows. First, we replace A and B by their centered versions. Let $A^{\frac{1}{2}}$ be the square root of A, and $S = A + A^{-\frac{1}{2}} B B^T A^{-\frac{1}{2}}$. Diagonalize S as $S = U_s \Lambda_s U_s^T$. Then $\tilde{K}$ is diagonalized by:

$$V = \begin{bmatrix} A \\ B^T \end{bmatrix} A^{-\frac{1}{2}} U_s \Lambda_s^{-\frac{1}{2}}$$

Then we have $\tilde{K} = V \Lambda_s V^T$ and $V^T V = I$. Given this decomposition of $\tilde{K}$ we proceed as usual for kPCA, by normalizing the eigenvectors Λ_s and projecting $\tilde{K}$ onto the normalized eigenvectors. This gives a dimensionality reduction of our images that makes the discrimination task easier.

4. Clustering

We view our collection as a semi-supervised dataset with errors that we wish to "clean up". First we form discriminants from faces with only one common extracted name. While this is a fairly small set of faces and the labels are not perfect, they let us form an initial discriminant space. We project all of our images into this space and perform a modified k-means procedure for clustering. This gives a larger and less noisy dataset from which to recompute discriminants. These new discriminants give a better representation of identity and we use them to re-cluster. This gives a reliable set of clusters.

4.1 Modified K-Means Clustering

Each image has an associated vector, given by the kPCA and LDA processes, and a set of extracted names (those words extracted using our proper name detector from the image's caption).

The clustering process works as follows: 1. Randomly assign each image to one of its extracted names 2. For each distinct name (cluster), calculate the mean of image vectors assigned to that name. 3. Reassign each image to the closest mean of its extracted names. 4. Repeat 2-3 until convergence (i.e. no image changes names during an iteration)

4.2 Pruning Clusters

We use a nearest neighbor model to describe our dataset and throw out points that have a low probability under this model. We remove clusters with fewer than three images so that nearest neighbor has some meaning. This leaves 19,355 images. We then remove points with low likelihood for a variety of thresholds (table 2) to get error rates as low as 5%, (error rates are calculated by hand labelling pictures from our dataset). We define likelihood as the ratio of the probability that it came from its assigned cluster over the probability that it did not come from its assigned cluster:

$$Likelihood(x) = \frac{P(x|c_i)}{P(x|\neg c_i)} \approx \frac{k_i}{k} log_2 \left(\frac{(n - n_i)}{n_i} \right)$$

where for a point x in cluster c_i, k is the number of nearest neighbors we are considering, k_i is the number of those neighbors that are in c_i, n is the total number of points in the set and n_i is the number of points in cluster c_i. We are using $log_2(\frac{n_i}{n})$ as the estimated probability of a cluster. This gives more weight to smaller clusters.

4.3 Merging Clusters

We would like to merge clusters with different names that actually correspond to a single person such as *Defense Donald Rumsfeld* and *Donald Rumsfeld* or *Venezuelan President Hugo Chavez* and *Hugo Chavez*. This can be extremely hard to do directly from text, in situations such as *Colin Powell*

#Images	#Clusters	error rate
19355	2357	26%
7901	1510	11%
4545	765	5.2%
3920	725	7.5%
2417	328	6.6%

Table 2: *Dataset sizes, number of clusters and error rates for different thresholds. The number of people in each set is approximately the number of clusters and error rate is defined as: given an individual and a face from their associated cluster, the error rate is the probability of that face being incorrect. We see that for mid-level pruning we get quite small error rates. We have turned a large semi-supervised set of faces into a well supervised set and produced clean clusters of many people.*

and *Secretary of State* (the names do not share any words). We propose to merge names that correspond to faces that look the same. Our system automatically propose merges between clusters that have similar compositions i.e. if the clusters have similar faces in them they possibly describe the same person. We can judge the similarity of clusters by the distance between their means in discriminant coordinates. Table 1 shows that almost all the top proposed merges correspond to correct merges (two names that refer to the same person), except that of *Richard Myers* and *President Bush*.

5. Quantitative Evaluation

We view this method as taking a supervised dataset that contains errors and ambiguities in the supervisory signal and producing a dataset with an improved (or, ideally, correct) supervisory signal, possibly omitting some data items. We must now determine how much better the new dataset is. Two things can have happened: First, the dataset may be smaller. Second, the supervisory signal should be more accurate.

But how much more accurate? If we are willing to assume that all errors are equivalent, we can see this issue as a coding problem. In particular, one must determine how many bits need to be supplied to make the dataset correct. We compute this score on a per-item basis, so that the size of the dataset does not affect the score. The number computed is necessarily an estimate — we may not have an optimal coding scheme — but if one uses a reasonably competent coding scheme should allow us to rank methods against one another.

5.1 The cost of correcting unclustered data

We have a set of *image-text pairs*, each of which consists of an image of one of the individuals, and a list of between 1 and 4 (typically) text labels. For each data item, we must now determine whether this set of labels contains the correct label. Letting C represent the random variable that takes on values of c or $\neg c$, we can do this with $H(C|r)$ bits (where $H(C|r)$ is the conditional entropy of C given r). If the list of r labels contains the correct name, we can tell which it is with $\log r$ bits. If it does not, we must supply the correct name, which will cost $H(L)$ bits (the entropy of the label set).

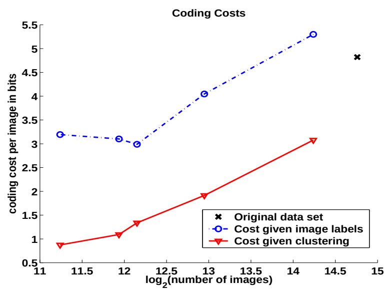


Figure 4: *The figure shows the approximate cost per item of correcting each dataset plotted against the size of the dataset. Note that the original dataset (the cross) is large and noisy; if one clusters, merges and cleans, then ignores the clustering structure, the resulting dataset is, in fact, somewhat noisier (dashed line). Finally, if one looks at the cluster structure, too, the dataset is much cleaner in particular one is contributing information to a dataset by clustering it correctly. Finally, increasing settings of our reject threshold leads to datasets that tend to be smaller and cleaner.*

Write $p(c|r)$ for the proportion of images with r labels that have a correct label in that list, and $p(\neg c|r)$ for the proportion of those with r labels that do not. The total cost *per item* of correcting the original dataset $\mathcal{D}$ in this fashion is then

$$C(\mathcal{D}) = \sum_{r=1}^{R} p(r)[H(C|r) + p(c|r)\log r + p(\neg c|r)H(L)].$$

5.2 The Cost of Correcting Clustered Data

Our clustering procedure introduces two types of structure. First, it assigns each image from an image-text pair to one of M clusters. Second, it associates a text to each *cluster*, so implicitly labelling each image pair within that cluster. If the labelling represented in this way is perfect, no further bits need to be provided.

We compute the *additional cost for perfect labelling* by examining how much work (how many bits) are required to fix all the errors in an imperfect clustering and cluster labelling. We assume the entropy of the label set remains fixed. We split the problem of fixing up imperfect clusters into two steps. In the first step, we change the name of each cluster, if necessary, so that it corresponds to the person who appears most frequently in this cluster. In the second step, we change the label of each image in a cluster if it does not correspond to the (now correct) label of the cluster.

Fixing the cluster labels: Let $p(d)$ be the proportion of clusters with the correct label and $p(\neg d)$ the proportion of clusters with an incorrect label. Let D be the random variable representing whether a cluster is labelled correctly or not and recall our practice of writing the entropy of a random variable X as $H(X)$ The cost *per cluster* of correcting the labels for all the clusters is then

$$H(D) + p(\neg d)H(L).$$

Fixing incorrect elements within the cluster: We assume for simplicity that, once the labels have been corrected as above, the proportion of correctly labelled elements in a cluster $p(e)$ is independent of the cluster. This means that the cost per item of fixing incorrectly labelled elements is independent of the specific cluster structure. Then to finish fixing the labelling we must pay $H(E)$ bits per item to identify the incorrect items and $C(D)$ bits per incorrect item to obtain the correct name, and the cost is

$$H(E) + p(\neg e)C(D)$$

Now write c for the total number of clusters per item (we hope $c < 1$). We have a total *per item* cost for the correct labelling of the data, after clustering, as

$$H(E) + p(\neg e)C(D) + c[H(D) + p(\neg d)H(L)].$$

Quantitative Evaluation Results

We report results for (a) the original dataset (b) the datasets resulting from our clustering, merging and cleaning process, without using cluster information (c) the datasets resulting from our clustering, merging and cleaning process, including their cluster structure. Figure 4 shows the plot. The original dataset is large and noisy; if one clusters, merges and cleans, then ignores the clustering structure, the resulting dataset is somewhat noisier. Finally, if one looks at the cluster structure, too, the dataset is much cleaner — this is because *one is contributing information to the dataset by clustering it correctly* and this fact is reflected by our score: in particular many of our clusters have the right face, but the wrong name (e.g. *President Daniel Arap Moi* in figure 3), and so can be corrected quickly. Finally, distinct settings of our reject threshold lead to datasets that tend to be smaller and cleaner.

References

[1] F.R. Bach, M.I. Jordan, "Kernel independent component analysis", *International Conference on Acoustics, Speech, and Signal Processing*, 2003

[2] K. Barnard,P. Duygulu, N. de Freitas, D.A. Forsyth, D. Blei, M.I. Jordan, "Matching Words and Pictures", *Journal of Machine Learning Research*, Vol 3, pp. 1107-1135, 2003.

[3] K. Barnard and P. Duygulu and D.A. Forsyth, "Clustering Art", *Computer Vision and Pattern Recognition*, Vol II, pp. 434-441, 2001.

[4] P. Belhumeur, J. Hespanha, D. Kriegman "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection" *Transactions on Pattern Analysis and Machine Intelligence*, Special issue on face recognition, pp. 711-720, July 1997.

[5] A.C. Berg, J. Malik, "Geometric Blur for Template Matching," *Computer Vision and Pattern Recognition,*Vol I, pp. 607-614, 2001.

[6] V. Blanz, T. Vetter, "Face Recognition Based on Fitting a 3D Morphable Model," *Transactions on Pattern Analysis and Machine Intelligence* Vol. 25 no.9, 2003.

[7] P. Duygulu, K. Barnard, N. de Freitas, D.A. Forsyth "Object Recognition as Machine Translation", *European Conference on Computer Vision*, Vol IV, pp. 97-112, 2002.

[8] J. Edwards, R. White, D.A. Forsyth, "Words and Pictures in the News," *Workshop on Learning Word Meaning from Non-Linguistic Data*, 2003.

[9] R. Fergus, P. Perona, A. Zisserman, "Object Class Recognition by Unsupervised Scale-Invariant Learning," *Computer Vision and Pattern Recognition*, 2003

[10] A.W. Fitzgibbon, A. Zisserman: "On Affine Invariant Clustering and Automatic Cast Listing in Movies," *European Conference on Computer Vision*, 2002

[11] C. Fowlkes, S. Belongie, F. Chung and J. Malik, "Spectral Grouping Using The Nyström Method," *TPAMI*, Vol. 26, No. 2, February 2004.

[12] R. Gross, J. Shi and J. Cohn, "Quo Vadis Face Recognition?," *Third Workshop on Empirical Evaluation Methods in Computer Vision*, December, 2001.

[13] R. Gross, I. Matthews, and S. Baker, "Appearance-Based Face Recognition and Light-Fields," *Transactions on Pattern Analysis and Machine Intelligence*, 2004.

[14] T. Leung, M.C. Burl, and P. Perona, "Finding Faces in Cluttered Scenes using Random Labelled Graph Matching", *Int. Conf Computer Vision*, 1995.

[15] K. Mikolajczyk "Face detector," *Ph.D report*, INRIA Rhone-Alpes

[16] D. Ramanan and D. A. Forsyth, "Using Temporal Coherence to Build Models of Animals", *Int. Conference on Computer Vision*, 2003.

[17] J. Scheeres, "Airport face scanner failed", *Wired News*, 2002. http://www.wired.com/news/privacy/0,1848,52563,00.html.

[18] C. Schmid, "Constructing models for content-based image retrieval", *Computer Vision and Pattern Recognition*, 2001.

[19] L. Sirovitch, M. Kirby, "Low-dimensional procedure for the characterization of human faces", *J. Opt. Soc. Am.* Vol 2, pp. 586-591, 1987.

[20] B. Scholkopf, A. Smola, K.-R. Muller "Nonlinear Component Analysis as a Kernel Eigenvalue Problem" *Neural Computation*, Vol. 10, pp. 1299-1319, 1998.

[21] M. Turk, A. Pentland, "Face Recognition using Eigenfaces", *Computer Vision and Pattern Recognition*, pp. 586-591, 1991.

[22] C. Williams, M. Seeger "Using the Nyström Method to Speed up Kernel Machines," *Advances in Neural Information Processing Systems*, Vol 13, pp. 682-688, 2001.

[23] M.H. Yang, N. Ahuja, D. Kriegman, " Face Recognition Using Kernel Eigenfaces," *Int. Conf. on Image Processing*, vol. 1, pp. 37-40, 2000

[24] W. Zhao, R. Chellapa, and A. Krishnaswamy, "Discriminant analysis of principal components for face recognition," *International Conference on Automatic Face and Gesture Recognition*, pp. 336-341, 1998.

[25] "WordNet, a lexical database for the English Language", 2003.