AutoAlbum: Clustering Digital Photographs using Probabilistic Model Merging*

John C. Platt
Microsoft Research
1 Microsoft Way
Redmond, WA 98052, USA
jplatt@microsoft.com

Abstract

Consumers need help finding digital photographs in their personal collections. AutoAlbum helps users find their photos by automatically clustering photos into albums. The albums are presented in an easy-to-use browsing user interface. AutoAlbum uses the time and order of photo creation to assist in clustering: albums consist of temporally contiguous photos. The content-based clustering algorithm is best-first probabilistic model merging, which is fast and yields clusters that are often semantically meaningful.

Keywords: digital photography, AutoAlbum, image browsing, image clustering, model merging

1. Introduction

It is getting increasingly popular for consumers to buy digital cameras and take thousands of photos of daily life. Most consumers simply dump these photos into one directory, analogous to dumping developed prints into a shoebox. A typical user generates thousands of photos a year. Finding a photo in this shoebox directory is difficult.

One tool to help find photos is a keyword search of image annotations. However, most consumers will not annotate their own images. Another tool is image retrieval [10], which can help a user find an image similar to an existing seed image. With a tablet [5] or a sketching interface [11], image retrieval can also be used to find images even when the user does not have a seed image.

A *browsing user interface* provides an alternative method for a user to find a photo which requires neither a sketch nor a seed image [1, 2, 6]. In such an interface, a number of photos are displayed to a user, who selects one or more images by clicking a mouse button. The system responds to these mouse clicks by displaying more likely images. The process repeats until the user finds the desired photo.

AutoAlbum is a browsing user interface that is explicitly designed for consumer digital photography. It is further described in Section 2. AutoAlbum clusters images into meaningful albums using best-first model merging, which is related to agglomerative clustering and is described in Section 3. Results of AutoAlbum are presented in Section 4.

1.1. Previous Work in Image Browsing

The crudest browsing interface simply shows thumbnails of every image in the shoebox directory. The user only is required to click once to fetch the desired image, but scanning through thousands of images is impractical.

At the other extreme is the deep tree search of PicHunter [2]. PicHunter shows four likely images at every step, so that the user scanning time is insignificant. However, the number of browsing steps can be large.

Similarity pyramids [1] prevent deep tree searches by organizing the entire database into a hierarchical quad-tree structure. Similarity pyramids use a fast agglomerative clustering algorithm to place similar images near each other at every level of the pyramid, and to ensure that an image at one level is similar to all of the images below it. Similarity is measured via image features.

2. AutoAlbum: From Images to Albums

Using image features to cluster consumer photos may not lead to desirable results. As the size of the shoebox directory gets large, the probability of matching two semantically dissimilar images becomes high. For example, an image of a sunset and an image of a fire truck may be placed into the same cluster. Therefore, a clustering algorithm may place semantically similar images into different clusters, making the search longer and more difficult.

This paper presents AutoAlbum, which is a combination of a clustering algorithm and a user interface that helps a

*To appear: Proc. IEEE Workshop on Content-Based Access of Image and Video Libraries, 2000 consumer find specific photos. AutoAlbum makes a tradeoff between generality and performance. Namely, AutoAlbum will be specific to the task of consumer digital photography. In return, the clustering performance will be much better, in both computation time and semantic performance.

AutoAlbum will cluster images into albums. Albums are very intuitive and desirable: users state that the most important feature of a photo organization tool is to automatically place photographs into albums [9].

A user will interact with the results of AutoAlbum in the following way. First, for every album, a representative thumbnail image is displayed. Therefore, the first user step is to search through all of the albums. The representative image reminds the user of the desired picture, either by image similarity, semantic similarity, or simply by being a memory aid. When the user has identified a likely album, he or she clicks on the representative image. AutoAlbum then shows thumbnails of all of the pictures in the album. The user then clicks on the desired image. Thus, AutoAlbum generates a two-level deep tree (see Figure 3). Figure 3 is purely schematic: the representative images and the images inside one album are presented in a two-dimensional matrix, similar to a photographic contact sheet.

2.1. Clustering Scenarios

In order to improve clustering beyond simply using image features, AutoAlbum uses metadata associated with digital photography. Depending on the camera and the download method, two different types of metadata for digital photographs are commonly preserved.

The first type of metadata is the creation time (and date) of the photo. Many digital cameras save the creation time along with the photo. Under some circumstances, this creation time is preserved as the data is downloaded from the camera. For example, directly reading the flash card of the Kodak DC260 preserves photo creation time.

The clustering algorithm for the first type of metadata only uses the creation time and ignores the content of the image. This is called *time-based clustering*. The clustering algorithm starts a new album if a new photo is taken more than a certain amount of time since the last photo. Clustering on the creation time works extremely well: temporally related pictures are almost always semantically related.

The second type of metadata is the order in which the photos are taken. This metadata is used when the camera, download method, or storage method does not preserve the photo creation time. For example, the camera clock may get reset due to dead batteries. Or, on some cameras, if the pictures are downloaded via a serial cable, the creation time of the file is the download time, not the true photographic creation time.

However, even when the creation time is destroyed, the order of the photos is very often preserved. The order can be deduced from the creation time of the file on a PC or from the file name of the image, which is sequentially generated.

In this more difficult second case, albums must be formed via clustering by content. However, the clustering should obey the order of the photos. A cluster should only contain a contiguous set of photos.

Exploiting the order of photos significantly helps clustering. In agglomerative clustering [3], the distances of an image I to all other images are considered. If the set of semantically related images to I is fixed as the size of the database grows, and if there is a constant probability that a given unrelated image is closer to image I than the closest related image, then the probability that the closest image to I is semantically related to I decays exponentially with database size. However, with ordered clustering, the distances from I to only two other images are measured. Therefore, the quality of the clustering should be independent of the database size.

For both time-based clustering and content-based clustering, the albums and the images within the albums should be displayed in the order that the photos were taken. This intuitive order corresponds to the order that people normally store their photographic prints [9].

Time-based and content-based clustering can also be combined. Time-based clustering can be used first. If the creation time is preserved, then small semantic clusters will be produced. Otherwise, large clusters will be produced. These large clusters can be further broken down by content-based clustering into smaller clusters.

3. Clustering via Best-First Model Merging

AutoAlbum uses probabilistic clustering to get the best content-based clustering performance. Probabilistic clustering creates a conditional density model of the image data, where the probability of generating an image depends conditionally upon the cluster membership of the image. In this case, the clustering metric is the likelihood of the data being generated by the model. In other words, AutoAlbum uses a maximum likelihood formulation.

The probabilistic model used is a Left-Right Hidden Markov Model (HMM). Album membership is a hidden state variable. For every state, there is a generative model for images in that state. This generative model is content-based: it is a probability of generating the pixels of the image, conditioned on the state. Also for every state, there is a probability of transitioning to the next state when presented with a new photo. This HMM model makes the reasonable assumption that consumer photographs can be described as piecewise stationary: a user's photographs can be grouped into contiguous clusters, each with similar image characteristics. If $\mathcal{D}$ is the image data, $\mathcal{A}$ is the album assignment,

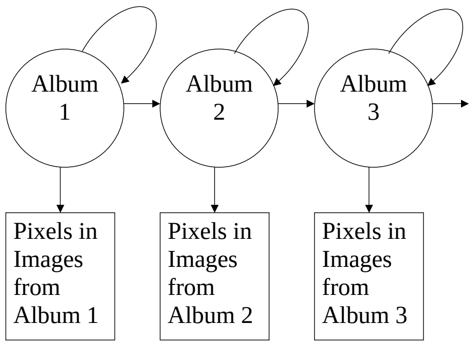
Album 1
Album 2
Album 3
Pixels in Images from Album 1
Pixels in Images from Album 2
Pixels in Images from Album 3

Figure 1. AutoAlbum assumes a Hidden Markov Model of photograph generation

and $\mathcal{M}$ is the HMM model for the photographs, then

$$P(\mathcal{D}, \mathcal{A}|\mathcal{M}) = P(\mathcal{D}|\mathcal{A}, \mathcal{M})P(\mathcal{A}|\mathcal{M}) \tag{1}$$

The image generation model $P(\mathcal{D}|\mathcal{A}, \mathcal{M})$ is further described in Section 3.1.

The Baum-Welch algorithm [8] can be used to fit an HMM to a shoebox directory of photos. However, there is a severe problem with this method. As applied to AutoAlbum, the Baum-Welch algorithm starts with an initial assignment of photos to albums and improves the assignment until a local maximum is reached. In practice, Baum-Welch only makes minor changes to the initial assignment, which produces unusable albums.

Best-first model merging [7, 12] helps avoid the problem of local maxima by stepwise merging adjacent clusters. Best-first model merging is similar to agglomerative clustering [3], except that it is probabilistic and works on an ordered data set.

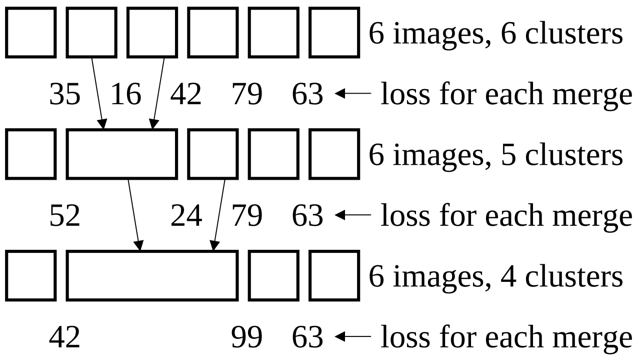
6 images, 6 clusters
35 16 42 79 63 ← loss for each merge
6 images, 5 clusters
52 24 79 63 ← loss for each merge
6 images, 4 clusters
42 99 63 ← loss for each merge

Figure 2. Best-First Model Merging

As applied to AutoAlbum, best-first model merging starts with every image having its own model. Merging a pair of adjacent models causes a loss of data likelihood, because one combined model is more general and cannot fit the data as well as two individual models. If L_X is the log likelihood of all of the pixels assigned to an album X given an associated model of X, and if albums X and Y are being merged to form album Z, then the change in log likelihood associated with the merging is

$$\Delta L = L_Z - L_X - L_Y. \tag{2}$$

The log likelihood of an album is defined in (3). Note that the change in $P(\mathcal{A}|\mathcal{M})$ due to a merge is tiny compared to the change in $P(\mathcal{D}|\mathcal{A}, \mathcal{M})$, hence is ignored. Best-first merging greedily merges the two adjacent models that cause the least loss of data likelihood.

The merging is repeated until a desired number of clusters is reached. For example, an average of 8 pictures per album may be desirable, so the number of clusters can be 1/8 the number of pictures. Alternatively, Bayesian techniques may be used to stop the model merging, as in [12].

For high-speed implementation, Omohundro [7] suggests storing all possible merges in a priority queue. One step of best-first model merging with a priority queue only requires $O(\log N)$ time, where N is the current number of clusters. Clustering of 405 images into 60 albums requires 0.34 CPU seconds on a 266 MHz Pentium II (not counting time to compute the image features).

3.1. AutoAlbum's Image Generation Model

AutoAlbum assumes that each HMM state generates the colors of the images in the corresponding album. The generated colors are at lower resolution than the original photo. The luminance of the images is assumed to come from lighting conditions, rather than from the state variable. This model was chosen after empirically testing several different models: it was the model which gave a content-based clustering that most closely matched the time-based clustering on the same data.

The colors are defined in the 1976 CIE $u'v'$ perceptually uniform color space [4]. The generative model is a histogram: $u'v'$ space is divided into 256 square bins (16 bins on a side). The model is that a bin is chosen with probability proportional to the stored histogram count and the color of a pixel is the color of the center of the chosen bin. There are 16 bins in u' that span from 0.1612 to 0.2883. There are 16 bins in v' that span from 0.4361 to 0.5361. Colors outside of these spans are clipped to the nearest bin.

During the best-first model merging, only the histogram statistics for each album is kept, to increase the speed of the clustering. The log likelihood of the pixels in album X given the 256 histogram counts X_i is given by

$$L_X = \sum_i X_i \log\left(\frac{X_i}{\sum_j X_j}\right). \tag{3}$$

When two histograms are merged, the counts in each corresponding bin are summed together.

At the first stage of best-first merging, each image has its own histogram model. Each histogram is initialized with all bins equal to a small value (10/256), which implies that the histogram estimate will be a *maximum a posteriori* estimate, and that the prior probability for this estimate is a uniform prior over all colors. Then, the image is downsampled by the lowest integer scale so that the final width is near 80 pixels and the final height is near 64 pixels. The RGB pixel values are converted to $u'v'$.

The histogram is further smoothed with a kernel. For every $u'v'$ pixel value in the downsampled image, a bilinear tent-shaped kernel is placed over the pixel value in $u'v'$ space. This kernel is two bin widths wide in each dimension. The histogram bins are then increased by an amount equal to the value of the kernel at the center of each bin. For example, if a pixel value lies directly on a bin center, that bin is increased by 1, and no other bins are changed. If a pixel value lies in the exact center of four bin centers, all four of those bin centers are increased by 0.25. Thus, the histogram counts are bilinearly interpolated. This interpolation is used to make the histogram model insensitive to small changes in color: slight changes to a pixel alters the model only slightly.

4. Experimental Results

To test AutoAlbum, data was gathered from two photographers (R and P) over a period of time. The creation times of the photographs were preserved. R took 1320 photos over the 12 months, while P took 405 photos over 4 months. The first 294 photographs of R have a corrupted creation time, which makes the R data set challenging.

To provide a “ground truth” for AutoAlbum, the author hand-clustered the data sets into contiguous semantic albums. Time-based clustering, content-based clustering, and a combined clustering algorithm are applied to the set. The time-based clustering used a threshold of one hour. The content-based clustering ignored the time of the photo and only used the order of photo creation. Content-based clustering was halted at the same number of clusters as the time-based clustering: 60 for P and 106 for R. The combined clustering started with the time-based clusters, and split any cluster equal to or larger than 48 images into content-based clusters with an average size of eight.

The effectiveness of AutoAlbum is measured via the F1 metric. Every image is considered a query. The clustered album that contains the query is considered the query result, while the human-generated album that contains the query is considered the truth. The true positives, false positives, and false negatives are microaveraged over all images, and the F1 score is then computed. An F1 score of 100% indicates a clustering that completely matches the human judgement.

The F1 scores for AutoAlbum are shown in Table 1. As a baseline, the images were clustered into 60 equal-sized albums for P and 106 equal-sized albums for R. The F1 scores for this simple algorithm are also shown in Table 1. For comparison, the results for all of these methods are shown for the subset of the R dataset that has non-corrupt creation times (the R1 column).

	Photographer		
	P	R	R1
Time clustering	**96.1%**	35.8%	61.9%
Content clustering	58.3%	64.5%	**67.6%**
Combined clustering	**96.1%**	**65.1%**	65.4%
Equal-sized clusters	40.2%	51.0%	50.4%

Table 1. F1 Clustering Performance

Table 1 shows that both time and content clustering work well in different situations. For the P data set, time clustering is very close to the human-selected albums. For the R data set, the time is corrupted, so time clustering is worse than baseline. However, combining the time and content clustering works well for P, R, and R1.

Qualitatively, the content-based clustering of AutoAlbum works very well. In Figure 3, content-based clustering separates photos corresponding to the outside of a house, a party, furniture in a living room, chairs in a hallway, and outdoor shots of buildings.

The representative photo from each album is simply the photo in the center of the ordered cluster. Another way of generating representative photos is to choose the image with the highest $P(\mathcal{D}|\mathcal{A},\mathcal{M})$. More testing is required to optimize the choice of representative photo.

The user interface of automatically generated albums is very intuitive. Examples of these albums are available at http://research.microsoft.com/∼jplatt/autoAlbum/ex.html

5. Conclusions

In summary, AutoAlbum automatically clusters consumer digital photographs into albums. The user only needs to search a small number of albums, select a likely album, then search the photographs within the album. The albums are generated by clustering, either by time and/or by content. The content-based clustering is accomplished via best-first probabilistic model merging, which forms clusters out of temporally contiguous photographs. Even though the clustering metric is simple, the resulting albums are frequently semantically meaningful.

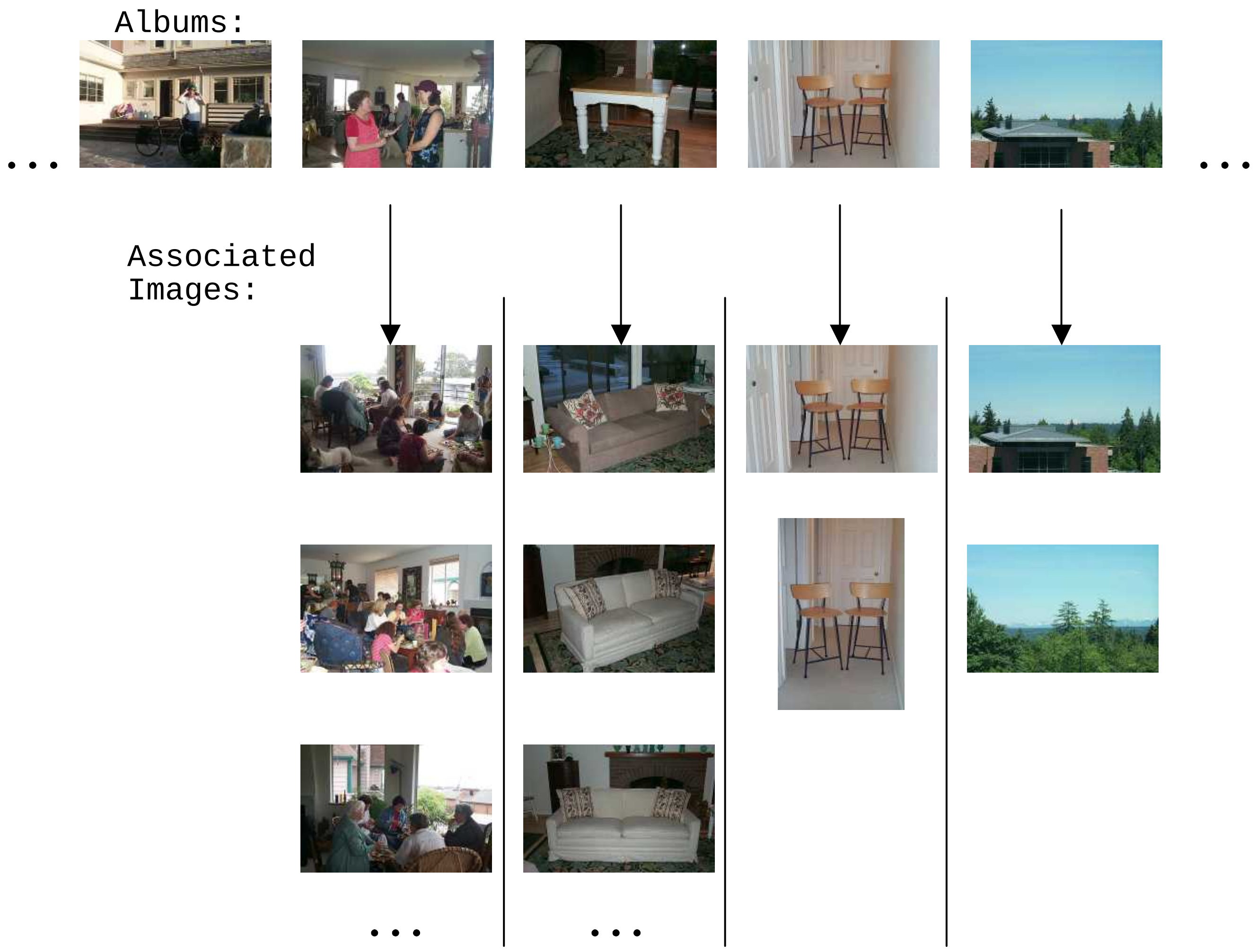

Figure 3. AutoAlbum Creates a Two-Level Hierarchy of Albums and Images

References

[1] J. Chen, A. Bouman, and J. C. Dalton. Similarity pyramids for browsing and organization of large image databases. In *Proc. SPIE/IS&T Conf. on Human Vision and Electronic Imaging III*, volume 3299, pages 563–575, 1998.

[2] I. J. Cox, M. L. Miller, S. M. Omohundro, and P. N. Yianilos. PicHunter: Bayesian relevance feedback for image retrieval. In *Proc. ICPR*, 1996.

[3] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.

[4] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley, 2nd edition, 1990.

[5] C. Jacobs, A. Finkelstein, and D. Salesin. Fast multiresolution image querying. In *Proc. ACM Siggraph 95 Conf.*, pages 277–286, 1995.

[6] A. Kuchinsky, C. Pering, M. L. Creech, D. Freeze, B. Serra, and J. Gwizdka. FotoFile: a consumer multimedia organization and retrieval system. In *Proc. ACM CHI Conf.*, pages 496–503, 1999.

[7] S. M. Omohundro. Best-first model merging for dynamic learning and recognition. In *Advances in Neural Information Processing Systems*, volume 4, pages 958–969, 1992.

[8] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77(2):257–286, 1989.

[9] K. Rodden. How do people organise their photographs? In *BCS IRSG 21st Ann. Colloq. on Info. Retrieval Research*, 1999.

[10] Y. Rui, T. S. Huang, and S.-F. Chang. Image retrieval: Current techniques, present directions, and open issues. *J. Visual Communications and Image Representation*, 10:39–62, 1999.

[11] J. R. Smith and S.-F. Chang. VisualSEEk: a fully automated content-based image query system. In *Proc. ACM Multimedia 96*, pages 87–98, 1996.

[12] A. Stolcke and S. Omohundro. Hidden markov model induction by bayesian model merging. In *Advances in Neu-*

ral Information Processing Systems, volume 5, pages 11–18, 1993.