Memory Constrained Face Recognition

Ashish Kapoor, Simon Baker, Sumit Basu and Eric Horvitz
Microsoft Research
{akapoor,sbaker,sumitb,horvitz}@microsoft.com

Abstract

Real-time recognition may be limited by scarce memory and computing resources for performing classification. Although, prior research has addressed the problem of training classifiers with limited data and computation, few efforts have tackled the problem of memory constraints on recognition. We explore methods that can guide the allocation of limited storage resources for classifying streaming data so as to maximize discriminatory power. We focus on computation of the expected value of information with nearest neighbor classifiers for online face recognition. Experiments on real-world datasets show the effectiveness and power of the approach. The methods provide a principled approach to vision under bounded resources, and have immediate application to enhancing recognition capabilities in consumer devices with limited memory.

1. Introduction

Face recognition has become a commodity application. A variety of applications such as Google Picasa, Adobe Photoshop Elements, Apple iPhoto, Facebook, and Microsoft Windows Live Photo Gallery use face recognition to help users with tagging their photos. The ubiquity of these tools highlights the success of face recognition—and also the considerable computational power now available on commodity personal computers. However challenges still remain for face recognition on mobile devices, precisely where matching faces to identities might deliver great value, given how cumbersome tagging can be in mobile settings. Despite the growing commonality of fast processors and high resolution cameras, even high-end phones such as the Apple iPhone 4S are currently limited to 512MB of RAM. Given that the OS and multiple applications must run at the same time, any individual application is typically limited to a few tens of MBs. The data structures for computing features of images, and for performing other tasks such as face detection, may consume a significant portion of this space, often leaving a surprisingly small amount of memory to store the classifier.

To understand why memory limitations make face recognition difficult, we must first examine the operation

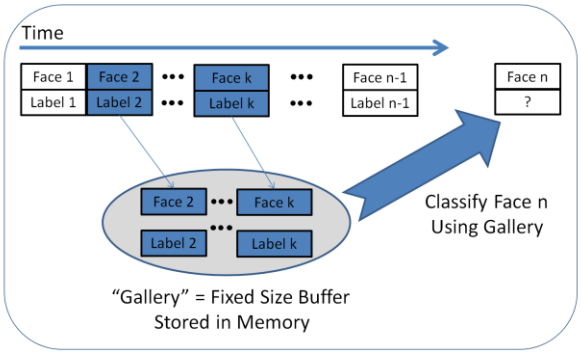


Figure 1. Streaming version of the face recognition problem, where a user repeatedly captures photos and uses face recognition to help tag other images. For each face in the sequence, previous faces can be used in training. We pursue methods that identify the best subset of previously captured faces to retain.

of effective face recognition methods. In recent years, the top performing algorithms have followed the approach of learning a feature set and a distance metric between face instances and then applying the nearest neighbor (NN) rule to determine the identity of an unlabeled face. This approach is used primarily because face recognition tends to be massively multi-class; a device owner can easily have hundreds of friends and family members in their photos. Further, the NN methodology can easily incorporate novel classes in the framework as they are encountered without requiring any further augmentation or training. Other parametric methods such as SVMs or decision trees need to be augmented upon discovery of new classes and can perform poorly in light of large numbers of target classes.

The NN procedure is simple and efficient when memory is not an issue. However, it quickly becomes impractical under constrained memory as we are unable to store the full set of examples. The inability to retain all of the data makes many other schemes harder to apply as well, since we cannot optimize over the entire set. The problem can be even more difficult: new classes (faces) might show up at any time. We thus have an unknown and constantly growing number of classes, coupled with an inability to store all past examples. We refer to this as the *streaming recognition* problem (see Figure 1).

We shall address challenges with leveraging streaming recognition for recognizing faces. Our main contribution is a method for selecting the best set of exemplars to store in memory, inspired by the decision-theoretic notion of expected value of information (EVI). The method continues to estimate the value of storing exemplars for classification by considering the discriminative potential of each data, given what the system *expects to see* in the future. The approach provides coherent principles for recognition under bounded memory in the setting of streaming data. We focus on the challenge of streaming face recognition as both an illustrative and valuable real-world example.

2. Related Work

Recent approaches in face recognition [1][3][34], have been shown to perform well on large databases. However, these methods have not been applied within conditions of scarce memory, nor for streaming recognition challenges.

Studies of machine learning under constrained memory have been done in areas outside of face recognition. The most promising and general methods take the popular kernel learning methodology for classifiers to the streaming or *online* scenario. The work of Crammer et al. [5] sought to reduce the memory footprint of these methods by adaptively discarding examples that were deemed to be less useful. Dekel et al. [8] followed this approach with greater formalism, presenting bounds on the cost incurred by such approximation schemes, and also developing methods that considered a fixed budget of examples. We were drawn to the latter line of work for its elegance and relevance to the streaming scenario; unfortunately, the domain we tackle is multiclass in nature and must consider uncertainty about the (often large) number of classes at hand. As the prior online learning work applies to binary classifiers, it is not necessary to keep one-vs.-all classifiers for each class, which would not be tenable in our scenario.

Given a multiclass, metric space scenario, nearest neighbor methods are an attractive option; it is no surprise that prior work in face recognition has found success with this approach. There has been prior research on nearest neighbor algorithms under constrained resources. Such efforts began with the seminal Condensed Nearest Neighbor (CNN) work of Peter Hart in 1968 [16], which incrementally grows the training data to the minimal "consistent set," i.e., the set required to correctly classify all training examples. Reduced Nearest Neighbor (RNN) [12] works from the other direction, iteratively removing examples that don't affect the training error. Modern approaches have included Modified Condensed Nearest Neighbor (MCNN) [9], which is order independent, and more recently, Fast Condensed Nearest Neighbor (FCNN) [2], which takes a similar approach to MCNN but is faster and produces smaller consistent sets. Although these and related methods perform admirably, they all require multiple passes through the data. So, despite their promise for static datasets or for offline training of static classifiers, they do not address the streaming recognition scenario, involving online decisions about which data to keep.

While systems researchers have not been concerned with nearest neighbor *per se*, the problem of caching in memory, disk, and internet access has led to a variety of algorithms for deciding which instances of memory to keep versus discard. Since access of a given cell often implies future access of its neighbors, the problems are similar to the ones we address. An excellent survey of modern techniques can be found in [33]. Our approach is similar in spirit to the classic cache-replacement algorithms from the computer systems literature, with extensions from machine learning and decision making. In particular, we draw upon ideas in active learning aimed at triaging effort in labeling examples. Past criteria for such triaging include disagreement among a committee of classifiers [10], margins of binary classifier [32], uncertainty [4][17], and expected informativeness [28], as well as the value of information [24]. In computer vision, active learning has been employed for object categorization [17] [21], video annotation [35], and face tagging [25]. Our work is different from these approaches in two ways: first, for every data point observed, the label is automatically revealed after the classification. Second, we operate in a streaming setting, while most of the work in active learning focuses on the pool-based setting, where all the data (though not the labels) are known beforehand.

Our work is inspired by research in the realm of decision-theoretic approaches to bounded rationality [18][19], and most similar to the work on active learning that considers the value of acquiring and storing data used by a classifier over the lifetime of a system [22][23][24].

3. Face Recognition with Limited Memory

Consider the setting where a stream of face patches is encountered. At each time step, a recognition system attempts to classify the observed face using the existing nearest-neighbor classification model. These face images then are presented to the user, who then has the opportunity to either accept the classification or correct the label. The system then updates its nearest neighbor classification model based on the user feedback. When there are no constraints on memory space, the optimal solution is to store the entire history. However, storing the entire stream of encountered face exemplars is not only infeasible but also expensive to search efficiently at classification time. Consequently, we aim to sample the stream selectively and store only those exemplars that would be most valuable for the purpose of future classifications.

Formally, we denote the stream of exemplars as $\{x_1, .., x_T\}$. Let us assume that we have a buffer B, which consists of tagged face images encountered in the

past and is limited in size to contain at most $MAXBUFFER$ number of examples. At each time step t, the system classifies the encountered exemplar x_t by first finding the nearest neighbor in the buffer B and then assigning the corresponding label to the new point. After the classification, the true label y_t is revealed, and the system then has the opportunity to update the buffer. The limited size of the buffer forces us to only store a finite number of examples; thus, the problem can be formally defined as finding an optimal set that contains at most $MAXBUFFER$ number of examples that would provide best classification performance on future face images.

We aim to quantify the value of each exemplar for the purpose of future classification tasks and to use computations of these values as a guiding principle in deciding on the optimal set for the buffer. Intuitively, each exemplar provides some information about the classification task, which in turns varies from point to point according to the underlying data distribution. Besides the discriminative quality, the information value of each exemplar is also a function of the expected *number of times* that data point will be used in the future. For example, an exemplar near a decision boundary may have high information content but due to the dynamic nature of the streaming scenario, the exemplar may never be used for classification. Thus, we also need to consider the probability distribution over usage to compute the expected value of information (EVI).

Since we are tackling the streaming setting, the underlying data distribution is constantly changing; thus the challenge in computing EVI entails the tasks of modeling discriminative quality of each example and continuing to update the probability that the exemplar will be useful for future classification. We next focus on these two tasks.

3.1. Modeling Discriminative Quality

One of the key components in an EVI formulation is estimating how important an exemplar would be in a classification task. Due to the heterogeneous distribution of data, different exemplars provide different amounts of discriminatory power. We aim to estimate the discriminative quality of each point in the buffer B by considering its proximity to various test cases that the system encounters over its lifecycle. Since the classification system is based on a nearest neighbor scheme, we consider relative distances of the test point with the set of points in the buffer. Intuitively, if many test points are classified correctly because of their proximity to an exemplar, then we can consider that example to be relatively important.

Formally, assume that the system at time step t uses the buffer to classify x_t, the current test case at hand. We define the probability that the test point x_t would choose an exemplar x stored in the buffer as its nearest neighbor:

$$p^{NN}(x|x_t) = \frac{\exp[-\gamma\|x_t - x\|^2]}{\sum_{x_i \in B} \exp[-\gamma\|x_t - x_i\|^2]}$$

This formulation is derived from the distance-metric learning literature [14], where it has been used for learning distance measures in a nearest neighbor context. Note that this expression is essentially a softened version of the nearest neighbor rule where γ is a fixed constant determining the softness; as γ becomes large, the nearest exemplar dominates the measure, and it asymptotically approaches the nearest neighbor rule. We can use this softened probability to quantify the discriminative power of the examples in the buffer. A correct classification of test examples should provide appropriate contribution in the measurement of the discriminatory power of the examples.

We denote the informativeness at time t of a point x in the buffer as $Info_t(x)$. If the example x_t is classified correctly, then we can update our estimation of the discriminatory power using:

$$Info_t(x) = Info_{t-1}(x) + p^{NN}(x|x_t) \cdot \delta[y = y_t]$$

Here, $\delta[y = y_t]$ denotes the indicator function. Intuitively, for every correct classification this update provides a contribution based on the probability that the test point x_t would choose x stored in the buffer as its nearest neighbor. By continuously updating this quantity for all the examples in the buffer, the system estimates how useful each point in the buffer has been over its lifecycle. Note as the parameter γ becomes large, $p^{NN}(x|x_t)$ goes to one for the nearest neighbor in the buffer and zero for others. Thus, our measure of discriminative power in this limiting case reduces to the scheme of maintaining counts of how many times each exemplar in the buffer was used as the nearest neighbor for correct classification. While counting is one way to measure discriminating quality, our probabilistic formulation instead also takes into account the spatial distribution of exemplars in the buffer.

3.2. Modeling Probability of Usage

The EVI measure captures the usefulness of an exemplar over the *future lifetime* of the classifier. Consequently, in addition to the discriminative power, we also need to model how likely each exemplar is going to be useful over time. In photo streams, images of the same people tend to cluster together in time. Face IDs often occur in bursts because people tend to take multiple pictures each time they get the camera out. We shall verify this assumption empirically in the experiments section below. Hence, our model of usage needs to leverage the burstiness and clustering of face IDs in time. Formally, for each exemplar x, we model $p_t^{use}(k|x)$, which represents the probability at time-step t that the data point will be used exactly k times in future. We assume that $p_t^{use}(k|x)$ follows a Negative-Binomial (NB) distribution parameterized with α and β:

$$p_t^{use}(k|x) \sim NegBin(\alpha, \beta) = \binom{k+\alpha-1}{\alpha-1}\left(\frac{\beta}{\beta+1}\right)^{\alpha}\left(\frac{1}{\beta+1}\right)^{k}$$

Algorithm 1: Memory Constrained NN - Stream

Initialize: Buffer $B = \{\}$, Data Stream = $\{x_1, .., x_T\}$
For t = 1 to T
  $\hat{y}$=$NearestNeighborClassify(x_t, B)$
  Observe true label y_t

  If $\hat{y} \neq y_t$
    If $Size(B) \leq MAXBUFFER$
      $B = B \cup \{x_t, y_t\})$
    Else
      $r^* = \text{argmin}_{i \in B}\, EVI_t(x_i)$
      $B = \{B - \{x_{r^*}, y_{r^*}\}\} \cup \{x_t, y_t\})$
    End If
  End If
End For

This distribution is a natural choice for modeling counts and is a robust alternative to the more commonly used Poisson distribution [13]. Intuitively, if we consider the occurrence of a face ID as a Bernoulli trial with probability $1/(1+\beta)$, then the Negative-Binomial distribution provides us with $p_t^{use}(k|x)$, the probability that we will next observe k instances of the same ID before encountering α number of failures. Thus, the distribution enables us to account for bursty sequences in a stream of photos. Also, note that the data following the Negative-Binomial distribution can be thought of as Poisson observations with a rate λ following the Gamma density $\lambda \sim Gamma(\alpha, \beta)$. The Negative-Binomial distribution is useful in modeling discrete data that are more variable than would be expected from the Poisson model (see [13]).

In the approach, each point x_i maintains its own probability distribution with parameters (α_i, β_i) and when introduced in the buffer initialized to $(\alpha_0 = 0, \beta_0 = 1)$. We increment both α_i and β_i by 1 at each time step, unless the incoming point is correctly classified because it is the nearest neighbor to x_i. Incrementing the parameters affects the probability distribution by decreasing the probability of the Bernoulli trial $(1/(1+\beta))$ and allowing for a greater number of failures (α); this reduces the expectation of seeing similar data in future. However, in case the data is a nearest neighbor to the correctly labeled test point both α_i and β_i are re-initialized to (α_0, β_0), signifying that the model expects to see more of such data in future.

3.3. Value of Information of Exemplars

We can calculate the EVI of each exemplar x in the active buffer at a time-step t by considering the estimated discriminating potential $Info_t(x)$ at the time-step t and the likelihood that it will be useful in future. If the exemplar x is used for classification exactly k times in future, the estimated benefit obtained in terms of utility is $k \cdot Info_t(x)$. However, we do not know *a priori* the exact value of k. Instead, at time t, we only have the probability $p_t^{use}(k|x)$ of the example being useful exactly k number of times in future. Consequently we need to consider all possible values of k in order to compute the EVI. Formally,

$$EVI_t(x) = \sum_{k=0}^{\infty} k\, Info_t(x) \cdot p_t^{use}(k|x) = Info_t(x) \cdot \bar{k}$$

Here, $\bar{k}$ denotes the expected value of k under the distribution $p_t^{use}(k|x)$. $Info_t(x)$ is independent of k and can be moved out of the summation. Our goal is to maintain a set of exemplars in the buffer that would maximize the utility of the set for the purpose of classifying future cases. Thus, when the model needs to incorporate a new data point, the procedure dictates replacing the point with minimum EVI with the new exemplar. In particular, the replacement candidate is chosen according to this criterion:

$$r^* = \underset{i \in B}{\text{argmin}}\, EVI_t(x_i)$$

The overall scheme for the memory constrained selection procedure is shown in Algorithm 1. The algorithm maintains a buffer of active points that is used for the purpose of nearest neighbor classification. Given a misclassification, the buffer is updated and the method is guided by EVI to choose a replacement candidate. Note, that for a Negative-Binomial distribution with parameters (α, β), the mean $\bar{k}$ is computed as α/β, which can be computed relatively efficiently. We note that the work presented here makes a greedy assumption, where we replace a single data point at a time. This can be generalized by applying myopic approximations and look-ahead procedures [27] that consider the EVI for different sets of points.

4. Experiments

Associating identities of faces in a photo collection can enhance photo browsing and management. For e.g., such tagging makes it possible to efficiently access all images containing a particular family member. We consider ongoing tagging scenarios where a stream of face images is continually captured by a mobile device or a small camera. Face detection is then performed on the image, detected faces are automatically classified, and results are presented to the user. The user, if he wishes, can then either accept the proposed tags or correct them. Note that, there is no requirement that an 'oracle' needs to keep providing labels all of the time. The scenario we propose is a natural one, where the user has the option to provide feedback to the system if they wish, and where the system is capable of incorporating and benefitting from such feedback.

4.1. Face Processing Pipeline

For our base feature extraction and recognition engine, we follow the Learning-Based Descriptor approach of Cao et al. [3]. The first step of this procedure is to detect the loca-

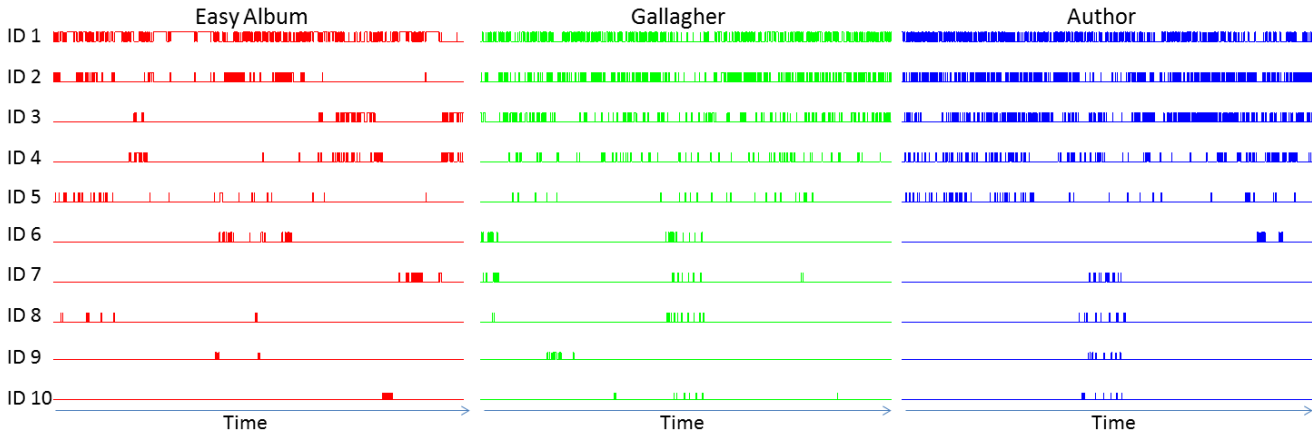


Figure 2. Illustration highlighting the bursty nature of labels. We plot the top ten most frequently observed labels in each of the three datasets. Rows correspond to occurrences of labels as they are encountered over time. The majority of labels tend to occur in clusters on the time axis.

tions of the eyes, nose, and mouth. Nine facial patches are then extracted from fixed positions relative to these patches and the resulting patches normalized for illumination variation by applying a DOG (Difference-of-Gaussian) filter. The learning-based descriptor is then computed by sampling the patches relative to the location of each pixel to create a feature vector that is then quantized using a learned encoding. The resulting patch code image is then compressed using PCA and normalized to form a descriptor. The patch descriptors are then concatenated and quantized into 16-bit integers. An L2 distance function is used. The final representation requires 800 bytes (2 bytes x 400 dimensions) for each face. This is roughly five times more efficient than the standard Local Binary Pattern approach (LBP) [1], yet obtains excellent rates on the Labeled Faces in the Wild (LFW) data [20].

4.2. Description of Data

In the streaming scenario, rather than having access to a gallery of images, the input is a sequence of face patches and the algorithm must attempt to classify each face using only the labels for faces encountered earlier in the sequence (modulo the memory buffer constraints). As the scenario differs from the classic face recognition task, the standard large face recognition databases such as Face Recognition Grand Challenge (FRGC) [31], Labeled Faces in the Wild (LFW) [20], or Multi-PIE [15] cannot be used. A key characteristic of this scenario, however, is that the performance of algorithms depends on the exact sequence of faces, and how coherent the IDs are across time.

To evaluate algorithms in the streaming scenario, it is important to use real sequences of photos taken from actual personal collections. We considered three such datasets[1]. Two of these are standard in the academic community. The third is a new dataset that we have created:

1. The EasyAlbum dataset [6] contains 1077 faces of 32 different people across 1044 photos.
2. The Gallagher dataset [11] contains 963 faces of 36 different people across 589 different images.
3. The Authors' dataset contains 1585 faces of 36 different people across 2032 photos.

In the creation of the third dataset, no selections of photographs were performed; the photos were simply copied from one of the author's devices. A face detection algorithm was run, and all faces that the author could name were labeled (*i.e.* a few irrelevant people in the background were not labeled, but otherwise there was no filtering to remove small, blurred, or low quality images).

4.3. Results

Burstiness Property: We first look at the statistics of the three datasets, specifically highlighting the bursty nature of labels. Figure 2 plots how labels across all the three datasets tend to cluster in time. We show the top 10 most frequently observed labels (ID 1 in Figure 2 is the top occurring label) and use an indicator function to plot their occurrence over the timeline. It is easy to see that labels do cluster in time. Although the most frequently occurring label (ID 1) is relatively uniform, a majority of other labels occur in bursts. This is observed across all three datasets, which were independently collected by different research groups. The visualization in Figure 2 provides strong evidence for modeling *probability of future use* using the model proposed in this paper.

Runtime Behavior: Figure 3 illustrates the runtime behavior of the method. Specifically, to generate Figure 3, we computed a 2D projection of the EasyAlbum data using t-SNE and then we plot the history of every data point in the active buffer (top). The bottom row shows slices through the time axis at 3 different points and captures the snapshot of the state of the system. All the points are color

[1] Data provided with supplementary material.

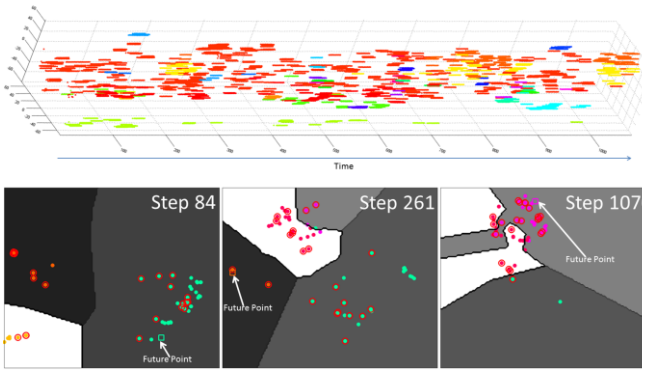


Figure 3. 2D projection of EasyAlbum data on a timeline showing how long each point was in the active buffer (top). Bottom row shows different time slices (color denotes labels) and red circle denote set of active points.

coded according to their labels. On the bottom row, we only plot the last 50 encountered points and the next point that needs to be classified. We observe that the active buffer evenly spans most of the set across all the time slices. Furthermore the bottom figure shows the flexibility that can be achieved by the proposed model. The method[2] is able to recognize the shift in data distribution (note the significant shift in labels from frame 84 to 1070), thereby successfully classifying future data points.

Recognition Performance: We compare the EVI method with several other schemes. Besides comparing against the simple scheme of random sampling, we also evaluate several alternate strategies motivated from caching. As discussed earlier, the problem of memory-constrained nearest neighbor classification in streaming scenarios has interesting parallels with the problem of maintaining a cache in the computer architecture and systems literature. While cache maintenance algorithms need to optimize for page faults, our goal is of maximizing correct classification. Nonetheless, we can modify some of the existing cache replacement policies to apply to the streaming photo scenario and then compare them with the EVI scheme. In particular, we consider the three popular cache replacement schemes of *least-recently-used (LRU), least-frequently-used (LFU) and first-in-first-out (FIFO)*. The least-recently-used and least-frequently-used methods need to maintain tables that keep counts about when and how many times respectively a face exemplar from the buffer has been used to correctly classify the stream of data. Upon misclassifying an exemplar, the least-recently-used heuristic would choose to replace the face from the buffer that has least recently been used for any correct classification. Similarly, least-frequently-used would choose the exemplar that has the least number of counts towards correct classification.

[2] We provide illustrative animations in supplementary materials that highlight such dynamic shifts in data boundaries

All of these five schemes (EVI, LRU, LFU, FIFO and Random) are compared against the baseline when the system can retain full memory (Full). In particular, we present results in terms of percent increase in misclassification error as compared to the use of infinite memory (i.e.: $100 * (\#Correct - \#Correct_{Full})/\#Correct_{Full}$). We report results for both a *held-out stream* of data (random 30%) and the rest of the *online stream* that our algorithm is being applied to. The accuracy on the online stream closely reflects the realistic task of *not* tagging every face encountered, whereas accuracy on the held-out set reflects the performance on a set of data that is untouched and separate from the training phase.

At each time-step t, we classify the encountered face exemplar in the online stream using the current active buffer and compute the percentage increase in misclassification. We then consider the label for the exemplar revealed and use it to update the active buffer. The cumulative performance on the held-out data is computed assuming original positions of test cases in the stream (i.e., a held-out image is classified only if the algorithm has seen all the training faces that came before). Note that the held-out stream is untouched in advance of evaluation, kept out of training and updating the buffer, and is only used to measure the test performance. We fixed the value of $\gamma = 1$ in all the experiments. Further, all the experiments were performed 100 times by randomly splitting the data into the hold-out and the online stream of unlabeled face exemplars. The results include average results over these 100 trials and the standard error.

Figure 4 compares the proposed EVI method with the other schemes. We show the results where the buffer size is fixed to 5% of the length of the data stream. Similar results are obtained using other buffer sizes and are summarized in Figure 4. The top row shows the result on the online streaming data where the lines represent average percent increase in misclassification up to that point using the different methods. The mean is computed over the 100 runs and the standard error is shown as dotted lines. Note that we are plotting accuracy up to each time step; consequently the graphs need not be monotonic (*e.g.,* the arrival of new previously unseen people can result in a string of errors which can push the cumulative accuracy lower).

We observe that the EVI based approach significantly outperforms both the random as well as the re-purposed caching schemes. While both the first-in-first-out (FIFO) and least-recently-used (LRU) schemes potentially can model the varying distribution of incoming face exemplars, they still do not leverage information about the different value of exemplars in classification. Similarly, the least-frequently-used (LFU) scheme roughly models the informativeness but does not take into account the dynamic data distribution in the streaming scenario. The EVI method on the other hand considers *both* the informativeness as well as the change in data distribution.

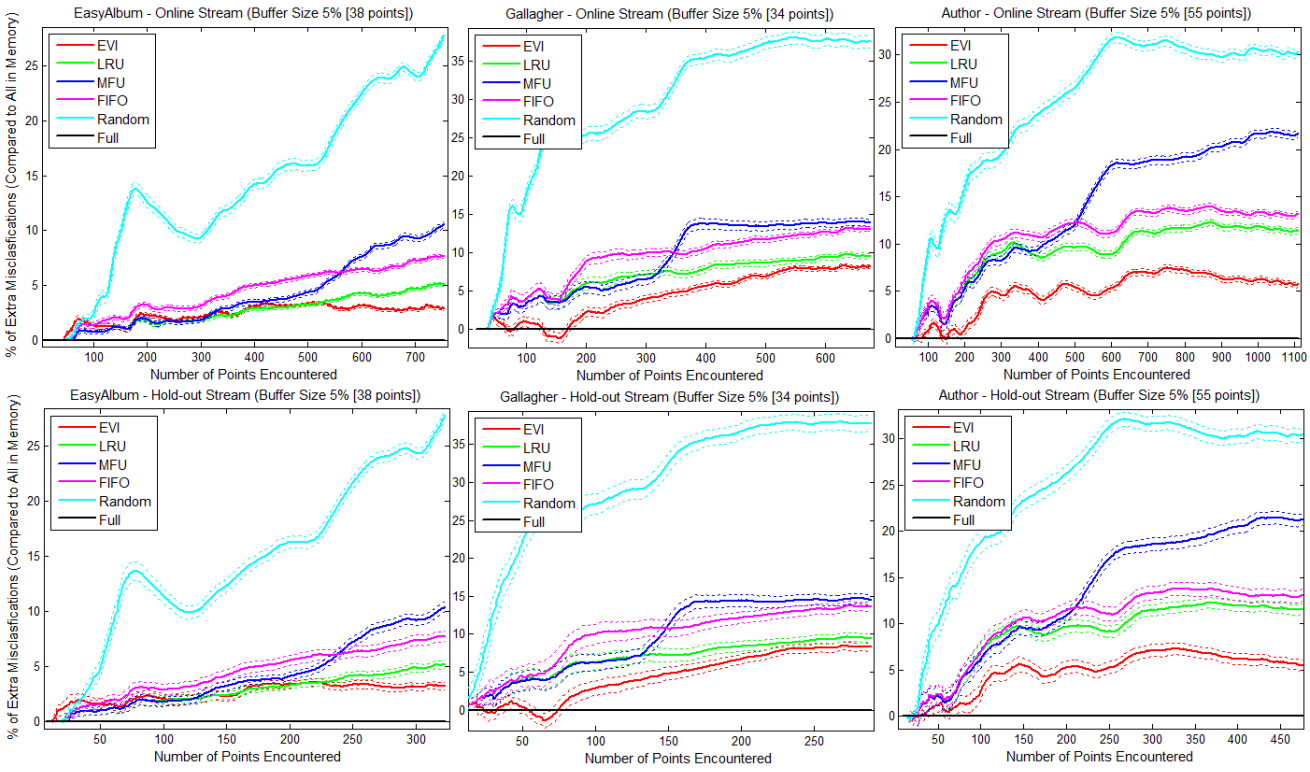


Figure 4. Comparison of EVI with other methods on EasyAlbum (left), Gallagher (center), and Author (right) data. (Top) Online performance on the streaming data. (Bottom) Performance on the held-out set. The graphs show mean increase in misclassification (100 runs) when compared to maintaining entire history (lower is better). Dotted lines signify std. error.

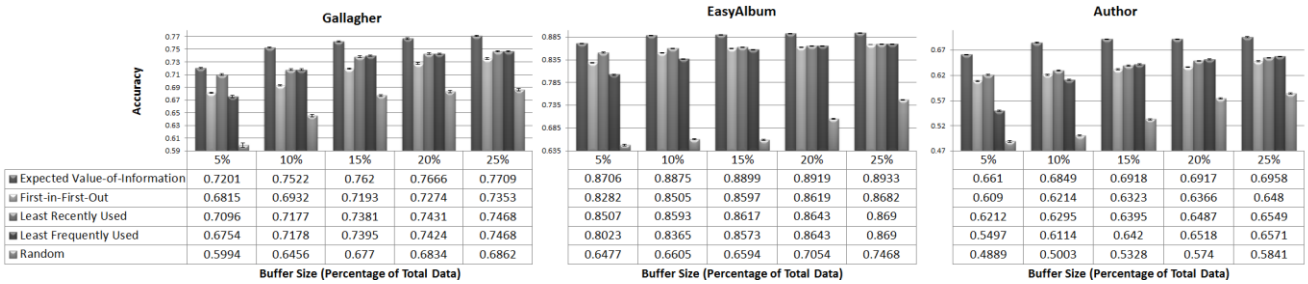
The results on the held-out stream of data are also shown in Figure 4 (bottom). We observe similar results confirming that the EVI method chooses better examples to keep, consequently achieving better performance than the other schemes. Finally, we note that the performance of the EVI scheme is only about by 5% worse than that of the system with infinite memory across all of the datasets.

Effect of Buffer Size: We also explore the performance of all five methods as we vary the active buffer size. Figure 5 shows the mean recognition accuracy estimated at the end of the entire online stream for different sizes. The average is again over 100 runs with the error bar denoting the standard error. We can observe that the recognition performance of the EVI method increases favorably with increase in buffer size. Increase in buffer size allows the algorithm to better model the dynamics of the data distribution as well as the ability of storing more exemplars in the active buffer resulting in better performance.

The experiments suggest that it is valuable to consider the expected value of information in memory-constrained classification—a common challenge in mobile computer vision applications. Perhaps the most important point to note about the results is how consistent they are across the 100 trials, the 3 datasets, the various buffer sizes, and the online and held-out streams. In all cases, a significant improvement is obtained by using our algorithm. Finally, we note that the proposed scheme based on expected value of information is efficient. In particular, each round for a stream of 963 exemplars (400 dims) took less than 1 msecs on a 2.16 Ghz Intel Laptop. Our implementation is in MATLAB and further speedups can be expected, making this scheme particularly appealing for small devices.

5. Conclusion and Future Work

We extend the nearest-neighbor classification paradigm to handle streaming recognition of faces under limited memory. The proposed scheme uses expected value of information to make decisions about which exemplars to store in a buffer based on the expected utility of each face patch over the future life of the system. The approach is applicable to numerous computer vision scenarios running in constrained memory. We highlighted the value of the methodology for enhancing a face-tagging application on mobile phones with limited memory. Experiments with real-world datasets demonstrate that the proposed framework is more effective than alternate baselines, and other methods motivated by cache replacement algorithms. A limitation of the current studies is that we have considered greedy schemes that replace one face at a time. We seek to study less myopic selective sampling methods. Other future directions include considering additional contextual cues to induce more constraints and applying the methodology to other streaming scenarios.



	5%	10%	15%	20%	25%
Expected Value-of-Information	0.7201	0.7522	0.762	0.7666	0.7709
First-in-First-Out	0.6815	0.6932	0.7193	0.7274	0.7353
Least Recently Used	0.7096	0.7177	0.7381	0.7431	0.7468
Least Frequently Used	0.6754	0.7178	0.7395	0.7424	0.7468
Random	0.5994	0.6456	0.677	0.6834	0.6862

	5%	10%	15%	20%	25%
Expected Value-of-Information	0.8706	0.8875	0.8899	0.8919	0.8933
First-in-First-Out	0.8282	0.8505	0.8597	0.8619	0.8682
Least Recently Used	0.8507	0.8593	0.8617	0.8643	0.869
Least Frequently Used	0.8023	0.8365	0.8573	0.8643	0.869
Random	0.6477	0.6605	0.6594	0.7054	0.7468

	5%	10%	15%	20%	25%
Expected Value-of-Information	0.661	0.6849	0.6918	0.6917	0.6958
First-in-First-Out	0.609	0.6214	0.6323	0.6366	0.648
Least Recently Used	0.6212	0.6295	0.6395	0.6487	0.6549
Least Frequently Used	0.5497	0.6114	0.642	0.6518	0.6571
Random	0.4889	0.5003	0.5328	0.574	0.5841

Figure 5. Mean recognition performance on all held-out cases at the end of the stream as size of the active buffer set is varied. We show results on Gallagher (left), EasyAlbum (center), and Author (right) data. Graphs show mean accuracy over 100 runs and the dotted lines are std. error, and show that EVI method is effective for a wide variety of buffer sizes.

References

[1] T. Ahonen, A. Hadid, and M. Pietikaninen. Face recognition with Local Binary Patterns. In *ECCV*, 2004.

[2] F. Anguilli. Fast condensed nearest neighbor rule. In *ICML,* 2005.

[3] Z. Cao, Q. Yin, X. Tang, and J. Sun. Face recognition with learning based descriptor. In *CVPR,* 2010.

[4] N. Cesa-Bianchi, A. Conconi and C. Gentile. Learning probabilistic linear-threshold classifiers via selective sampling. In *COLT,* 2003.

[5] K. Crammer, J. Kandola, and Y. Singer. Online classification on a budget. In *NIPS*, 2003.

[6] J. Cui, F. Wen, R. Xiao, Y. Tian, and X. Tang. EasyAlbum: An interactive photo annotation system baed on face clustering and re-ranking. In *SIGCHI*, 2007.

[7] E. Y. Chang, S. Tong, K. Goh, and C. Chang. Support vector machine concept-dependent active learning for image retrieval. In *IEEE Transactions on Multimedia*, 2005.

[8] O. Dekel, S. Shalev-Shwartz, and Y. Singer. The Forgetron: a kernel-based perceptron on a fixed budget. In *NIPS*, 2006.

[9] F. Devi and M. Murty. An incremental prototype set building technique. In *Pattern Recognition*, 35, 2002.

[10] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. In *Machine Learning*, 28, 1997.

[11] A.C. Gallagher. Clothing Co-segmentation for recognizing people. In *CVPR,* 2008.

[12] W. Gates. The reduced nearest neighbor rule. In *IEEE Trans. On Info. Theory*, 18, 1972.

[13] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. Bayesian Data Analysis. Chapman and Hall. 1998

[14] J Goldberger, S Roweis and G Hinton. Neighborhood Component Analysis. In *NIPS*, 2004.

[15] R. Gross, I. Matthews, J.Cohn, T. Kanade, and S. Baker. Multi-PIE. *Int. Conf. on Auto. Face & Gesture Reco.* 2008.

[16] P. Hart. The condensed nearest neighbor rule. In *IEEE Trans. On Information Theory,* 14, 1968.

[17] A. Holub, P. Perona, and M. Burl. Entropy-based active learning for object recognition. In *CVPR workshop on Online Learning for Classification*, 2008.

[18] E. Horvitz, Reasoning under varying and uncertain resource constraints. In *National Conference on Artificial Intelligence*, 1988.

[19] E. Horvitz and J. Lengyel. Perception, Attention, and Resources: A Decision-Theoretic Approach to Graphics Rendering. In *UAI*, 1997.

[20] G. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. *University of Massachusetts, Amherst, Technical Report 07-49,* 2007.

[21] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell. Active learning with Gaussian Processes for object categorization. In *ICCV*, 2007.

[22] A. Kapoor and E. Horvitz. Principles of Lifelong Learning for Predictive User Modeling. In *User Modeling*, 2007.

[23] A. Kapoor and E. Horvitz. On Discarding, Caching, and Recalling Samples in Active Learning. In *UAI* 2007.

[24] A. Kapoor, E. Horvitz, and S. Basu. Selective supervision: guiding supervised learning with decision-theoretic active learning. In *IJCAI.* 2007.

[25] A. Kapoor, G. Hua, A. Akbarzadeh and S. Baker. Which faces to tag: adding prior constraints into active learning. In *ICCV,* 2009.

[26] R. Karedla, J. S. Love, and B. G. Wherry. Caching strategies to improve disk system performance. In *IEEE Computer,* 27(3), March 1994.

[27] A. Krause, A. Singh, C. Guestrin. Near-optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. In *JMLR*, 2008.

[28] N. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process method: informative vector machines. In *NIPS*, 2002.

[29] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *International Conference on Research and Development in Information Retrieval*, 1994.

[30] N. Megiddo and D. S. Modha. ARC: A self-tuning, low overhead replacement cache. *FAST,* 2003.

[31] P.J. Philips, P.J. Flynn, W.T. Scruggs, K.W. Bowyer, J. Chang, K. Hoffman, J. Marques, J. Min, and W.J. Worek. Overview of the Face Recognition Grand Challenge. In *CVPR*, 2004.

[32] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. In *ICML*, 2000.

[33] J. Wang. A survey of web caching schemes for the internet. In *ACM SIGCOMM Computer Comm. Review,* 29(5), 1999.

[34] L. Wolf, T. Haussner, and Y. Taigman. Similarity scores based on background samples. In *ACCV*, 2009.

[35] R. Yan, J. Yang, and A. Hauptmann. Automatically labeling video data using multi-class active learning. In *ICCV*, 2003.