Multi-Class Active Learning by Uncertainty Sampling with Diversity Maximization

Article *in* International Journal of Computer Vision · June 2015

DOI: 10.1007/s11263-014-0781-x

Int J Comput Vis
DOI 10.1007/s11263-014-0781-x

Multi-Class Active Learning by Uncertainty Sampling with Diversity Maximization

Yi Yang · Zhigang Ma · Feiping Nie · Xiaojun Chang · Alexander G. Hauptmann

Received: 11 September 2013 / Accepted: 15 October 2014

Abstract As a way to relieve the tedious work of manual annotation, active learning plays important roles in many applications of visual concept recognition. In typical active learning scenarios, the number of labelled data in the seed set is usually small. However, most existing active learning algorithms only exploit the labelled data, which often suffers from over-fitting due to the small number of labelled examples. Besides, while much progress has been made in binary class active learning, little research attention has been focused on multi-class active learning. In this paper, we propose a semi-supervised batch mode multi-class active learning algorithm for visual concept recognition. Our algorithm exploits the whole active pool to evaluate the uncertainty of the data. Considering that uncertain data are always similar to each other, we propose to make the selected data as diverse as possible, for which we explicitly impose a diversity constraint on the objective function. As a multi-class active learning algorithm, our algorithm is able to exploit uncertainty across multiple classes. An efficient algorithm is used to optimize the objective function. Extensive experiments on action recognition, object classification, scene recognition, and event detection demonstrate its advantages.

Communicated by Kristen Grauman.

Y. Yang · X. Chang
Centre for Quantum Computation and Intelligent Systems,
University of Technology Sydney, Sydney, NSW, Australia
e-mail: yiyang@cs.cmu.edu

Z. Ma · A. G. Hauptmann
School of Computer Science, Carnegie Mellon University,
Pittsburgh, PA, USA

F. Nie (✉)
The Center for OPTical IMagery Analysis and Learning,
Northwestern Polytechnical University, Xi'an, China
e-mail: feipingnie@gmail.com

Keywords Active learning · Uncertainty sampling · Diversity maximization

1 Introduction

Typical visual concept recognition methods first train a classifier based on the labelled training data via a statistical approach, and then use the classifier to recognize visual concepts. In real-world applications, it is usually easy to obtain huge volumes of unlabelled data in an automatic way. However, a large number of labels are difficult to get, which require much human labour. Generally speaking, there are three types of approaches to relieve the tedious work of labelling the training data. The first one is known as semi-supervised learning, which combines both the labelled and unlabelled data to train the classifier for recognition (Zhu 2008). The second one is to borrow knowledge from related domain(s), such as transfer learning (Ma et al. 2014; Shen et al. 2014) and multi-task learning (Yang et al. 2013). On the other hand, to make the most use of the scarce human labelling resources, active learning selects the most informative data from a candidate set (usually referred to as active pool) for labelling. Instead of being a passive recipient of label information, the learning algorithm actively decides what data are more useful and then asks humans to label them for training.

As a different but complementary way to reduce the labelling cost in supervised learning, active learning has received much research attention. In recent years, researchers have proposed several active learning algorithms and applied them to different computer vision applications, e.g., image classification (Jain and Kapoor 2009; Joshi et al. 2009), concept detection (Li et al. 2010), object recognition (Gong et al. 2014), 3D reconstruction (Kowdle et al. 2011), tracking

Published online: 13 November 2014

(Vondrick and Ramanan 2011), correspondences mapping (Jegelka et al. 2014), etc. The key issue in active learning is how to decide whether a sample point is "useful" or "informative". For example, in (Chattopadhyay et al. 2012) it is realized by specifically selecting a set of query samples that minimize the difference in distribution between the labelled and the unlabelled data. In literature, representativeness sampling and uncertainty sampling are the two widely used criteria for selecting the training data to be labelled from the active pool. An uncertainty sampling active learning algorithm is usually associated with a classifier, which is used to evaluate the uncertainty of each data in the active pool. Despite the substantial progress made in uncertainty sampling, there are still several aspects to be improved.

First, as discussed in Jain and Kapoor (2009), most of the exiting research in active learning is based on binary classification classifiers. While relatively few approaches have been proposed for multi-class active learning, (e.g., Li et al. 2004; Yan et al. 2003), many of them are direct extensions of binary active learning methods to the multi-class scenario. However, many real world applications of visual concept recognition are multi-class problems. Decomposing a multi-class problem as several independent binary classification subproblems may degrade the performance of active learning. If we use a series of binary classifiers in active learning as those in Li et al. (2004), Yan et al. (2003) etc., the model is not able to evaluate the uncertainty of a sample across multiple classes. For example, if a sample is an uncertain sample for one class while it is a certain sample for another class, it is tricky for an algorithm to evaluate its uncertainty. Besides, given that the multiple binary classifiers are independent from each other, the algorithm cannot identify the classes that need more labelled training data (Jain and Kapoor 2009).

Second, uncertainty sampling algorithms tend to suffer from the problem of insufficient training data. Active learning algorithms usually start with a seed set, which contains only a small number of labelled data. Based on the seed set, a classifier is trained to evaluate the uncertainty of the candidate data in the active pool. The goal of active learning is to select the data to be labelled for training. Thus, at the beginning, the number of labelled data is very small, which is the nature of active learning. Performance of the classifier can be poor due to the small number of labelled data (Hoi et al. 2008; Yang et al. 2012). Based on SVM active learning (Tong and Chang 2001), Hoi et al. have proposed a min-max optimization algorithm to evaluate the informativeness of data points (Hoi et al. 2008), in which the unlabelled data are employed as complementary information. Compared with SVM active learning (Tong and Chang 2001), the min-max optimization algorithm is able to select training data in batch mode and is more robust to over-fitting. Empirical study shows that the min-max criterion proposed in Hoi et al. (2008) outperforms SVM active learning in Tong and Chang (2001). Hoi's algorithm calls QP solver to optimize the objective function, resulting in high computation complexity of $\mathcal{O}(n^3)$. In a later work (Hoi et al. 2009) Hoi et al. have proposed a solution to speed up the optimization approach, which makes the algorithm in Hoi et al. (2009) more applicable. Hoi's algorithm (Hoi et al. 2008, 2009) has improved the performance of active learning because it uses all the data in the active pool to evaluate the importance of each candidate. However, as the algorithm is based on the binary classifier SVM, it may become less effective when the data are multi-class.

最小最大可以用于batch-mode？

Motivated by the state of the art of active learning, particularly the semi-supervised active learning algorithm (Hoi et al. 2008, 2009), we propose a new multi-class active learning algorithm, namely Uncertainty Sampling with Diversity Maximization (USDM), which carefully addresses the small seed set problem by leveraging all the data in the active pool for uncertainty evaluation. Our algorithm is able to globally evaluate the informativeness of the pool data across multiple classes. Different from the other multi-class active learning algorithms e.g., (Jain and Kapoor 2009), our algorithm exploits all the active pool data to train the classifier, making the uncertainty evaluation more accurate. Further, most of the existing uncertainty sampling algorithms merely consider the uncertainty score for active learning, i.e., they select the active pool data which are closest to the classification boundaries. However, the data close to classification boundaries may be very similar to each other. If similar data are selected for supervision, the performance of active learning may degrade. In light of this, we propose to select the most uncertain data, which are as diverse as possible. It means that the data selected for labelling should be sufficiently different from each other. Compared to Jain and Kapoor (2009), USDM simultaneously utilizes both the labelled and unlabelled data in the active pool. While Hoi's algorithm (Hoi et al. 2008, 2009) exploits the entire active pool, the classifier embedded in USDM is more capable of evaluating uncertainty partially because it is a multi-class classifier and partially because it explicitly exploits the manifold structure of active pool. USDM has many merits, such as batch mode, multi-class, semi-supervised, efficient, and the diversity of selected data is explicitly guaranteed.

2 Related Work

In this section, we briefly review the related work. This paper is closely related to active learning and semi-supervised learning.

Active learning has been shown effective in many applications such as 3D reconstruction (Kowdle et al. 2011), and image retrieval (Wang et al. 2003). Existing active learning algorithms can be roughly divided into two categories which are representativeness sampling and uncertainty sampling. As it is important to exploit the data distribution when selecting the data to be labelled (Cohn et al. 1996), representativeness sampling tries to select the most representative data points according to data distribution. A typical way of this kind of approach is clustering based active learning, which employs a certain clustering algorithm to exploit the data distribution and evaluate representativeness. The performance of these algorithms directly depends on the clustering algorithm. Clustering algorithms are unsupervised and only converge to local optima, whose results may deviate severely from the true labels. It remains unclear how the clustering based algorithms will perform when the clustering is not sufficiently accurate. The other well-known approach of representativeness sampling is optimal experiment design (Yu et al. 2006). Based on optimal experiment design, a variety of active learning algorithms have been proposed, in which different Laplacian matrices have been utilized, e.g., (He et al. 2007). A limitation of optimal experiment design is that the optimization of the objective function is usually NP-hard. Certain relaxation is required. Then semi-definite programming (SDP) or sequential method (usually a greedy method) is applied to the optimization. However, SDP has high computational complexity and the greedy methods may converge to severe local optima.

Uncertainty sampling, also known as classifier based sampling (Campbell et al. 2000; Li and Sethi 2006), is the most frequently adopted strategy in active learning, which builds upon the notions of uncertainty in classification (Jain and Kapoor 2009). This type of algorithm is usually associated with a particular classification algorithm. A classifier is trained by a seed set consisting of a small number of randomly selected data. Data points in the active pool, which are most likely to be misclassified by the classifier, are regarded as the most informative ones. For example, support vector machine (SVM) active learning (Tong and Chang 2001) selects the data points which are closest to the classification boundary of the SVM classifier as the training data. In Wang et al. (2003), the transductive SVM classifier is used for active learning. Hoi et al have proposed to integrate semi-supervised learning and support vector machines for active learning and have achieved promising results on image retrieval (Hoi and Lyu 2005). In Brinker (2003), diversity constraint is combined with SVM for active learning. The uncertainty sampling strategy has also been combined with other classifiers, such as the Gaussian process (Kapoor et al. 2010), the K-nearest neighbor classifier (Lindenbaum et al. 2004) and the probabilistic K-nearest neighbor classifier (Jain and Kapoor 2009).

Semi-supervised learning has been widely applied to many applications with the appealing feature that it can use both labelled and unlabelled data (Yang et al. 2012; Zhu 2008). For instance, Zhu et al have proposed to utilize a Gaussian random field model with a weighted graph representing labelled and unlabelled data for semi-supervised learning (Zhu et al. 2003). Han et al. have proposed to use spline regression for semi-supervised feature selection (Han et al. 2014). In Hoi et al. (2008), the researchers have formulated the semi-supervised active learning algorithm as a min-max optimization problem for image retrieval. A semi-supervised learning based relevance feedback algorithm is proposed in Yang et al. (2012) for multimedia retrieval. The benefit of utilizing semi-supervised learning is that we can save human labor cost for labelling a large amount of data because it can exploit unlabelled data to learn the data structure. Thus, the human labelling cost and accuracy are both considered, which gives semi-supervised learning a great potential to boost the learning performance when properly designed.

The rest of this paper is organized as follows. In Sect. 3, we give the objective function of the proposed USDM algorithm. An efficient algorithm is described in Sect. 4 to optimize the objective function, followed by detailed experiments in Sect. 5. Lastly, we conclude this paper in Sect. 6.

3 Uncertainty Sampling with Diversity Maximization

In this section, we give the proposed USDM active learning algorithm. We start with discussing the approach for evaluating uncertainty of each sample. n is the total number of the data in the seed set and the active pool. Suppose there are n_s data in the seed set and n_p data in the active pool. It turns out that $n_s + n_p = n$. We are going to select m $(m < n_p)$ data for supervision. Denote $x_i \in \mathbb{R}^d$ as a sample which is either in the active pool or the seed set, where d is the dimension of the sample. To better utilize the distribution of the pool data and the seed set, we propose to evaluate the uncertainty via random walks on a graph (Zhu 2008). To begin with, we first construct a graph G, which consists of n nodes, one for each sample in the active pool or the seed set. The edge between the two nodes x_i and x_j is defined as follows.

$$W_{ij} = \begin{cases} \exp\left(\frac{-\|x_i - x_j\|^2}{\sigma^2}\right) & x_i \text{ and } x_j \text{ are } k\text{-nearest neighbors;} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Note that one can also define the unweighted edge between x_i and x_j as:

$$W_{ij} = \begin{cases} 1 & x_i \text{ and } x_j \text{ are } k\text{-nearest neighbors;} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

We take each vertex in the graph as a state in a Markov chain, i.e., each state corresponds to one sample in the seed set or the active pool. For the ease of representation, we define a diagonal matrix $D \in \mathbb{R}^{n \times n}$ whose element $D_{ii} = \sum_j W_{ij}$. Denote

$$Q = D^{-1}W, \tag{3}$$

which is partitioned into 2 × 2 blocks:

$$Q = \begin{bmatrix} Q_{ss} & Q_{ps}^T \\ Q_{sp}^T & Q_{pp} \end{bmatrix}, \tag{4}$$

where $Q_{ss} \in \mathbb{R}^{n_s \times n_s}$ denotes the normalized weight between the data in the seed set, $Q_{sp} \in \mathbb{R}^{n_s \times n_p}$ denotes the weight between the data from the seed set and the active pool; and $Q_{pp} \in \mathbb{R}^{n_p \times n_p}$ denotes the normalized weight between the data in the active pool. For the data in the seed set, we set the corresponding states as absorbing states, which only transit to themselves with the possibility of 1. If a sample x_i is in the active pool, it is a non-absorbing state. The one step transition probability from x_i to x_j is $T_{ij} = \frac{W_{ij}}{\sum_j W_{ij}}$. Then the transition matrix T of the Markov random walks with absorbing states is defined as

$$T = \begin{bmatrix} I_{n_s} & \mathbf{0}_{n_p} \\ Q_{sp}^T & Q_{pp} \end{bmatrix}, \tag{5}$$

in which $I_{n_s} \in \mathbb{R}^{n_s \times n_s}$ is an identity matrix, and $\mathbf{0}_{n_p} \in \mathbb{R}^{n_s \times n_p}$ is a matrix of all zeros. We use the calligraphy uppercase letters to represent set. Denote $\mathcal{P}$ as the active pool and $\mathcal{S}$ as the seed set. As demonstrated in Doyle and Shell (1984), the probabilities that the pool data are absorbed by the seed set data in equilibrium with transition matrix T is

$$P(\mathcal{S}|\mathcal{P}) = (I_{n_p} - Q_{pp})^{-1} Q_{sp}^T, \tag{6}$$

where $I_{n_p} \in \mathbb{R}^{n_p \times n_p}$ is an identity matrix. Define $Y_j = [Y_{1j}, Y_{2j}, \ldots, Y_{n_s j}]^T \in \{0, 1\}^{n_s \times 1}$ as the label indicator vector of the seed set for the j-th class. If $x_i \in \mathcal{S}$ belongs to the j-th class, $Y_{ij} = 1$; otherwise $Y_{ij} = 0$. Given a pool sample $x_t \in \mathcal{P}$, we define the probability that x_t is absorbed by the j-th class as the sum of the probabilities that it is absorbed by all the seed set data which are from the j-th class $\mathcal{C}_j$. The probabilities that the pool data are absorbed by the seed set data belonging to the j-th class can be formulated as

$$P(\mathcal{C}_j|\mathcal{P}) = (I_{n_p} - Q_{pp})^{-1} Q_{sp}^T Y_j. \tag{7}$$

The above procedure can be interpreted as a Dirichlet problem to get harmonic functions (Zhu 2008; Zhu et al. 2003). We define $F \in \mathbb{R}^{n \times c}$ as follows

$$F_{ij} = \begin{cases} (I_{n_p} - Q_{pp})^{-1} Q_{sp}^T Y_j & \text{if } x_i \in \mathcal{P}; \\ Y_{ij} & \text{if } x_i \in \mathcal{S}. \end{cases} \tag{8}$$

where c is the number of classes. It can be verified that $\sum_{j=1}^c F_{ij} = 1$. F_{ij} is regarded as the probability that x_i belongs to the j-th class, i.e., $P(\mathcal{C}_j|x_i) = F_{ij}$. For a sample $x_i \in \mathcal{P}$, we assume that its label can be estimated by a random variable ℓ_i. As Shannon Entroy is a natural choice to measure the uncertainty of random variables, we adopt the entropy $\mathrm{H}(\ell_i)$ to evaluate the uncertainty of x_i, which can be estimated by

$$\mathrm{H}(\ell_i) = -\sum_j^c P(\mathcal{C}_j|x_i) log\left(P(\mathcal{C}_j|x_i)\right), \tag{9}$$

where $log(\cdot)$ is the natural logarithm operator. A larger $\mathrm{H}(\ell_i)$ indicates that x_i is more uncertain. Denote f_i as the ranking score of x_i. The pool data with higher ranking scores are selected before the others for supervision. According to the uncertainty principle, we have the following objective function

$$\max_{\sum_i f_i = 1, f_i \geq 0} \sum_{x_i \in \mathcal{P}} -f_i \times \sum_j^c P(C_j|x_i) log\left(P(C_j|x_i)\right) \\ - \Omega(f_i) \tag{10}$$

which can be rewritten as

$$\max_{\sum_i f_i = 1, f_i \geq 0} \sum_{x_i \in \mathcal{P}} - \left[f_i \times \sum_j^c F_{ij} log(F_{ij}) \right] - \Omega(f_i), \tag{11}$$

In (10), the term $\Omega(f_i)$ is a function on f encoding the data distribution information, in other words, the diversity criterion in decision making. The constraint $\sum_{i=1}^n f_i = 1$ in the above function is imposed to avoid arbitrary scaling on f_i. Denote

$$b_i = \begin{cases} \sum_j^c F_{ij} log(F_{ij}) & \text{if } x_i \in \mathcal{P}; \\ 0 & \text{if } x_i \in \mathcal{S}. \end{cases} \tag{12}$$

Then we can rewrite (11) as

$$\min_{f_i} \sum\nolimits_{i=1}^n \frac{1}{|\log(1/c)|} (f_i \times b_i) + \Omega(f_i), \\ s.t. \sum\nolimits_{i=1}^n f_i = 1, f_i \geq 0. \tag{13}$$

In (13), the first term $\sum_{i=1}^n \frac{1}{|\log(1/c)|}(f_i \times b_i)$ is used to evaluate the uncertainty of the pool data. b_i is dependent on $F_{ij}, j \in \{1, \ldots, c\}$. Recall that $\sum_{j=1}^c F_{ij} = 1$, which means that F_{ij} is the probability such that x_i is in the j-th class. Thus the algorithm is able to estimate the uncertainty across multiple classes. If the uncertainty is measured by a binary classifier, the algorithm turns to a binary class active learning algorithm. In this sense, one main difference between our algorithm and the S-SVM active learning (Hoi et al. 2008) is that our algorithm is more capable of estimating the uncertainty of the data in an active pool, where the manifold structure is uncovered.

Based on (13), the problem is then how to define $\Omega(f_i)$ to incorporate the diversity maximization criterion. We propose a simple yet effective way of computing a kernel matrix $K \in \mathbb{R}^{n \times n}$. For example, if we use the well-known RBF kernel, the i, j-th element of K can be computed by $K_{ij} = \frac{-\|x_i - x_j\|^2}{\sigma^2}$, where σ is a parameter. Given two data x_i and x_j, if they are similar to each other, K_{ij} will have a large value. In this case, we shall not have the two data to be labelled simultaneously. In other words, if x_i is selected as a training sample, x_j should be excluded in some sense. Therefore, given that K_{ij} has a large value, at least one of f_i and f_j should have a small value. We then propose to minimize the following objective function to make the selected data as diverse as possible.

$$\min_{f_i} \Omega(f_i) = \min_{f_i} \sum_{i=1}^{n} \sum_{j=1}^{n} f_i f_j K_{ij}. \tag{14}$$

We can see that if K_{ij}, f_i and f_j are all large values, it will incur a heavy penalty on (14). Minimizing (14) makes the selected training data different from each other. Combining uncertainty criterion and diversity criterion, we have the following objective function for active learning.

$$\min_{f_i} \sum_{i=1}^{n} \left(\frac{r}{|\log(1/c)|} (f_i \times b_i) + \sum_{j=1}^{n} f_i f_j K_{ij} \right)$$
$$s.t. \sum_{i=1}^{n} f_i = 1, f_i \geq 0, \tag{15}$$

where r is a parameter. The objective function shown in (15) can also be viewed as a regularization framework for active learning, in which we use diversity constraint as a regularizer added to the traditional uncertainty sampling. The diversity regularization term is crucial because some of the uncertain data are potentially similar to each other. It is worth mentioning that the batch mode active learning task is a combinational optimization problem, as discussed in Hoi et al. (2009). The solution to (15) does not necessarily give the exact optimal solution to the batch mode active learning problem, where the goal is to exactly select an optimal set of the most informative examples. However, (15) approximates the optimal solution of the batch mode active learning in an efficient and effective way (Hoi et al. 2009). If we take a closer look at (15), it can be seen that this objective function is inspired by the algorithm proposed in Hoi et al. (2008, 2009). The major difference is the way how the algorithm does uncertainty estimation.

4 Efficient Optimization

In this section, we optimize the objective function of USDM. Let $f = [f_1, f_2, \ldots, f_n]^T$. For the ease of representation, we define $a = [a_1, a_2, \ldots, a_n]^T$, where $a_i = \frac{r \times b_i}{|\log(1/c)|}$ and r is a parameter. (15) can be rewritten as

$$\min_f f^T a + \frac{1}{2} f^T K f$$
$$s.t. \sum_{i=1}^{n} f_i = 1, f_i \geq 0. \tag{16}$$

The objective function shown in (16) is a standard quadratic programming (QP) problem, which can be readily solved by existing convex optimization packages. However, typical QP solver has a high computational complexity of $\mathcal{O}(n^3)$. It is more practical to make the optimization faster. In this section, we propose to use a faster algorithm to optimize the objective function (16), based on the augmented Lagrange multiplier (ALM) framework (Bertsekas 1999).

4.1 Brief Review of ALM

The ALM algorithm in Bertsekas (1999) is introduced to solve the following constrained minimization problem.

$$\min g(Z), \quad s.t. h(Z) = 0, \tag{17}$$

where $g : \mathbb{R}^d \to \mathbb{R}$ and $h : \mathbb{R}^d \to \mathbb{R}^{d'}$. A typical way to define the augmented Lagrangian function of (17) is

$$L(Z, U, \mu) = g(Z) + \langle U, h(Z) \rangle + \frac{\mu}{2} \|h(Z)\|_F^2, \tag{18}$$

where Z is the optimization variable, U is Lagrangian coefficient and μ is a scalar. The following procedure can be applied to optimizing the problem shown in (17).

Algorithm 1: General ALM method (Bertsekas 1999).

```
Set $\rho > 1, t = 1, U_1 = 0, \mu_1 > 0$;
repeat
    $\hat{Z} = \arg\min_Z L(Z, U_t, \mu_t)$;
    $U_{t+1} = U_t + \mu_t h(\hat{Z})$;
    $\mu_{t+1} = \rho \mu_t$
    $t = t + 1$;
until Convergence;
Output $\hat{Z}$.
```

4.2 Efficient Optimization of USDM

In this subsection, we introduce a fast optimization approach of our algorithm under the ALM framework (Bertsekas 1999; Delbos and Gilbert 2005). First we rewrite (16) as follows.

$$\min_{f,v} f^T a + \frac{1}{2} f^T K f$$
$$s.t. f^T \mathbf{1}_n = 1, v \geq 0, f = v, \tag{19}$$

where $\mathbf{1}_n \in \mathbb{R}^n$ is a vector of all ones. The augmented Lagrangian function of (19) is defined as

$$L(f, v, \mu, \lambda_1, \lambda_2) = \frac{\mu}{2} \left(f^T \mathbf{1}_n - 1 + \frac{1}{\mu} \lambda_1 \right)^2$$
$$+ \frac{\mu}{2} \left\| f - v + \frac{1}{\mu} \lambda_2 \right\|_F^2 + f^T a + \frac{1}{2} f^T K f$$
$$s.t. v \geq 0 \tag{20}$$

Note that

$$\min_f L(f, v, \mu, \lambda_1, \lambda_2) \Leftrightarrow \min_f \frac{1}{2} f^T A f - f^T e, \tag{21}$$

Algorithm 2: USDM active learning algorithm.

```
1  Initialization: set $\rho > 1$, $f_i = \frac{1}{n}$ $(1 \leq i \leq n)$, $v = f$, $\lambda_1 = 0$, and
   $\lambda_2 \in \mathbb{R}^n$ is a vector of all zeros, $\mu > 0$;
2  repeat
3      Update $A$ by $A = K + \mu I_n + \mu \mathbf{1}_n \mathbf{1}_n^T$;
4      Update $e$ by $e = \mu v + \mu \mathbf{1}_n - \lambda_1 \mathbf{1}_n - \lambda_2 - a$;
5      Compute $\hat{f}$ by solving the linear system $A\hat{f} = e$;
6      Compute $v$ by $v = \text{pos}(\hat{f} + \frac{1}{\mu}\lambda_2)$;
7      Update $\lambda_1$ by $\lambda_1 = \lambda_1 + \mu \times (\sum_{i=1}^{n} \hat{f}_i - 1)$;
8      Update $\lambda_2$ by $\lambda_2 = \lambda_2 + \mu \times (\hat{f} - v)$;
9      $\mu = \rho\mu$;
10 until Convergence;
11 Output $\hat{f}$.
```

where

$$A = K + \mu I_n + \mu \mathbf{1}_n \mathbf{1}_n^T \tag{22}$$

and

$$e = \mu v + \mu \mathbf{1}_n - \lambda_1 \mathbf{1}_n - \lambda_2 - a. \tag{23}$$

The objective function shown in (21) can be easily optimized by solving a linear system and we have

$$\hat{f} = \arg\min_{f} L(f, v, \mu, \lambda_1, \lambda_2) = A^{-1}e. \tag{24}$$

Meanwhile,

$$\begin{aligned} &\min_{v \geq 0} L(f, v, \mu, \lambda_1, \lambda_2) \\ \Leftrightarrow\ &\min_{v \geq 0} \left\| v - (f + \frac{1}{\mu}\lambda_2) \right\|^2 \end{aligned} \tag{25}$$

By solving the optimization problem shown above, we have

$$\hat{v} = \arg\min_{v \geq 0} L(f, v, \mu, \lambda_1, \lambda_2) = \text{pos}(q), \tag{26}$$

where $q = f + \frac{1}{\mu}\lambda_2$ and $\text{pos}(q)$ is a function which assigns 0 to each negative element of q, i.e., for any element $q_i \in q$, $\text{pos}(q_i) = \max(q_i, 0)$.

In summary, the proposed USDM active learning algorithm is listed in Algorithm 2 as shown above. It can be verified that Algorithm 2 converges to the *global optimum*. Except for step 5, the computation of all the steps in Algorithm 2 is very fast. It is worth noting that when computing $\hat{f}$ in step 5, we only need to solve a linear system. No matrix inversion is involved. Note that there are a few efficient linear system solvers that can be readily used. We may also use a faster algorithm to solve or approximate the linear system, e.g., (Spielman and Teng 2004). Because it is out of the scope of this paper, we omit the detailed discussion here.

Table 1 Dataset description

Name	Size	# of class	Application	Data type
KTH	2, 387	6	Action recognition	Video
Youtube	1, 596	11	Action recognition	Video
Coil	1, 440	20	Object classification	Image
Scene15	4, 485	15	Scene recognition	Image
MED	2, 874	18	Video event detection	Video

5 Experiment

In this section, we test the proposed active learning algorithm by applying it to a variety of visual concept recognition applications, including action recognition, object classification, scene recognition, and video event detection.

5.1 Experiment Setup

Five different public datasets are used in the experiment, which are KTH (Schüldt et al. 2004), Youtube (Liu et al. 2009), Coil (Nene et al. 1996), Scene15 (Lazebnik et al. 2006) and MED dataset collected by National Institute of Standards and Technology (NIST).[1] Table 1 summarizes the detailed information of the datasets used in the experiment.

We compare our algorithm to both representative sampling active learning and uncertainty sampling active learning algorithms in the experiment. The comparison algorithms include SVM active learning (SVM$_{active}$) proposed in Tong and Chang (2001), Semi-supervised SVM active learning (S-SVM) proposed in Hoi et al. (2008), Laplacian regularized optimal experiment design (LOED) proposed in He et al. (2007) and the multi-class active learning algorithm pKNN proposed in Jain and Kapoor (2009). The k used in our algorithm for graph construction is set to 5 empirically. For the parameters involved in different active learning algorithms, we similarly tune them from $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}, 10^{4}\}$ and report the best results.

Each dataset is randomly split into two non-overlapping subsets, one as training candidate set and the other as testing data set. In our experiment we fix the size of training candidate set as 1,000 for all the datasets. Denote c as the number of classes. First, we randomly select 3 positive samples for each class from the training candidate set, i.e., the size of the seed set is $3 \times c$. The remaining data in the training candidate set

[1] http://www.nist.gov/itl/iad/mig/

are regarded as the active pool. Then we run each active learning algorithm to select training data. We set the batch size as $1 \times c$, $2 \times c$,..., $7 \times c$, respectively. During the training we use both the data selected by the active learning and the data in the seed set as labelled samples. Therefore, there are $4 \times c$, $5 \times c$,..., $10 \times c$ labelled data for training. In our experiment, after the training data are selected by different active learning algorithms, a classifier is trained for visual concept recognition. The SVM classifier is used for SVM_{active}, S-SVM and LOED. For pKNN (Jain and Kapoor 2009), we use the classifier embedded in the algorithm (Jain and Kapoor 2009). For our USDM algorithm, we use the random walk classifier illustrated in Sect. 2 to generate the soft label representation for classification. As the random walk classifier is a transductive algorithm, after the pool data have been selected, we re-construct a larger graph including all data for testing data classification. Note that class labels of selected data could be imbalanced. Based on the soft label representation derived by the random walk, binary class labels for each class are then determined by an SVM.

5.2 Action Recognition

We use the KTH action dataset (Schüldt et al. 2004) and the Youtube action dataset (Liu et al. 2009) to compare the performance of different active learning algorithms in terms of action recognition. In this experiment, each video sequence is represented by 1,000 dimension BoW STIP feature (Laptev et al. 2008). KTH action dataset contains six types of human actions (walking, jogging, running, boxing, hand waving and hand clapping) performed several times by 25 subjects in four different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors (Schüldt et al. 2004). There are 2,387 action sequences in this dataset. The Youtube action dataset is a real-world dataset which was collected from Youtube. It contains intended camera motion, variations of the object scale, viewpoint, illumination as well as cluttered background. There are 11 actions in this data set which are basketball shooting, biking/cycling, diving, golf swinging, horseback riding, soccer juggling, swinging, tennis swinging, trampoline jumping, volleyball spiking, and walking with a dog (Liu et al. 2009). Four videos in Youtube action dataset are too short to be captured by the feature extracting code shared by Laptev et al. (2008), so we use a dataset of 1,596 sequences.

Figure 1 compares the performance of different active learning algorithms for action recognition using KTH dataset. We observe that LOED outperforms SVM_{active}. Meanwhile, S-SVM and pKNN significantly improve the accuracy, compared with SVM_{active} and LOED. A possible explanation is that SVM_{active} is vulnerable to over-fitting, because it does not consider the data distribution of the active pool during the learning process. S-SVM uses the unlabelled data and pKNN evaluates the uncertainty across multiple classes, and therefore more information is used in them. Our algorithm dramatically outperforms all the competitors at all batch sizes. Figure 2 shows the experimental results on action recognition using Youtube dataset. The video sequences in Youtube dataset were downloaded from Youtube. It is much more noisy than the lab-generated KTH dataset. Yet, we observe that our algorithm consistently outperforms other active learning algorithms. The experimental results demonstrate the advantages of our algorithm.

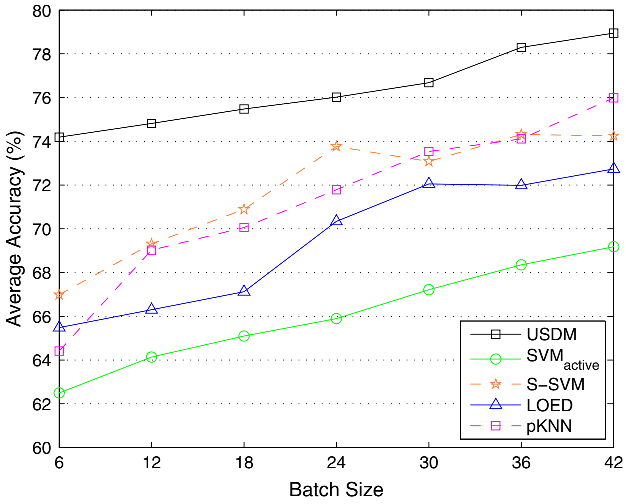


Fig. 1 A comparison of different active learning algorithms on action recognition using KTH dataset. There are 6 different actions in this dataset

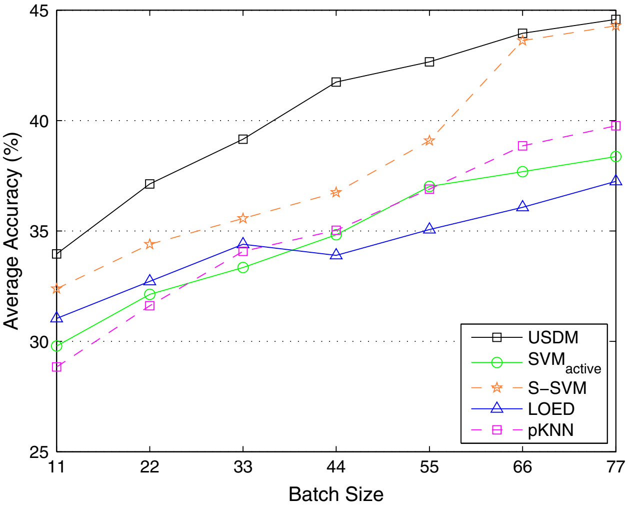


Fig. 2 A comparison of different active learning algorithms on action recognition using Youtube dataset. There are 11 different actions in this dataset

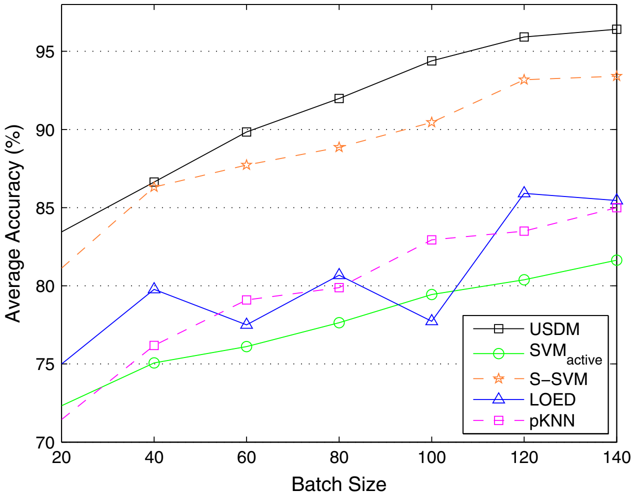


Fig. 3 A comparison of different active learning algorithms on object classification. There are 20 different objects in this dataset

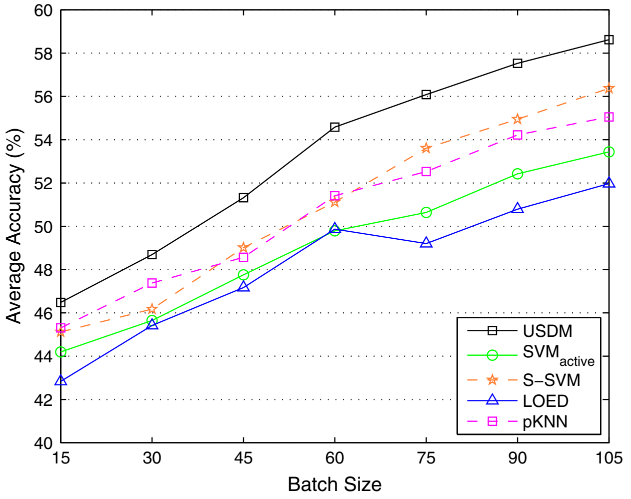


Fig. 4 A comparison of different active learning algorithms on scene recognition. There are 15 different scenes in this dataset

5.3 Object Classification

Figure 3 shows the experimental results of objective recognition on Coil dataset, which consists of 1,440 grey scale images (Nene et al. 1996). There are 20 different objects in total. Each image was resized to 32 × 32. We use the grey values as the features of the images, with dimension of 1024. Both pKNN and SVM$_{active}$ train classifiers only based on the seed set for uncertainty evaluation. We can see from Fig. 3 that pKNN generally outperforms SVM$_{active}$, indicating that it is beneficial to evaluate the uncertainty of data across multiple classes. S-SVM gains the second best performance due to the exploration on the data distribution of the active pool. Our algorithm achieves the best performance. Compared with the second best algorithm S-SVM, our algorithm has two main advantages. First, the random walk algorithm has better capability of uncovering the manifold structure (Tenenbaum et al. 2000) of the entire active pool to evaluate uncertainty of the pool data. Although S-SVM also takes the distribution of the pool data into consideration, the manifold structure is missed when training the SVM classifier for uncertainty evaluation. Thus, it somehow suffers from the small size of the training data, especially when the data has manifold distribution. Second, our algorithm is a multi-class active learning algorithm, which is able to evaluate the “informativeness” of the pool data globally.

5.4 Scene Recognition

To test the performance of USDM in scene recognition, we use the Scene15 dataset, which contains 4,485 images from 15 different scenes (Lazebnik et al. 2006). In this experiment, we extract HOG feature to represent the images and the dimension of the feature vectors is 6300.

Figure 4 shows the experimental results. Both pKNN and SVM$_{active}$ only use the seed set to train the classifiers for uncertainty evaluation. pKNN generally outperforms SVM$_{active}$, which indicates that multi-class active learning (e.g., pKNN) is a more powerful approach. If we take the pool data into consideration for training data selection, the performance will be further improved. As shown in Fig. 4, S-SVM outperforms SVM$_{active}$ at all batch sizes, but not as significantly as other applications. We observe that our algorithm outperforms all the other algorithms. Given that S-SVM gains good performance for this dataset and our USDM still outperforms S-SVM, we conclude that it is better to leverage the manifold structure of the active pool and the seed set to evaluate uncertainty for active learning.

5.5 Complex Event Detection

In this subsection, we compare the different active learning algorithms on complex event detection (Ma et al. 2014; Yang et al. 2013). We merge the MED10 dataset and the MED11 dataset into one, which is referred to as MED in this paper. In the experiment, we use the MoSift (Chen and Hauptmann 2009) descriptor, based on which a 32,768 dimension spatial BoW feature is computed to represent each video sequence (Yang et al. 2013). Principal component analysis is performed to remove the null space.

Figure 5 shows the keyframes from a video which is “changing a vehicle tire”. We can see that the MED dataset is rather “wild”. The problem is more difficult, compared to the other datasets. In the experiment, we have used all the videos which are labelled as one of the 18 events. The number of positive samples for each event varies from 80 to 170 and there are 2,874 positive samples for the 18 events in total.

Fig. 5 An example video sequence of "changing a vehicle tire" event from the MED dataset

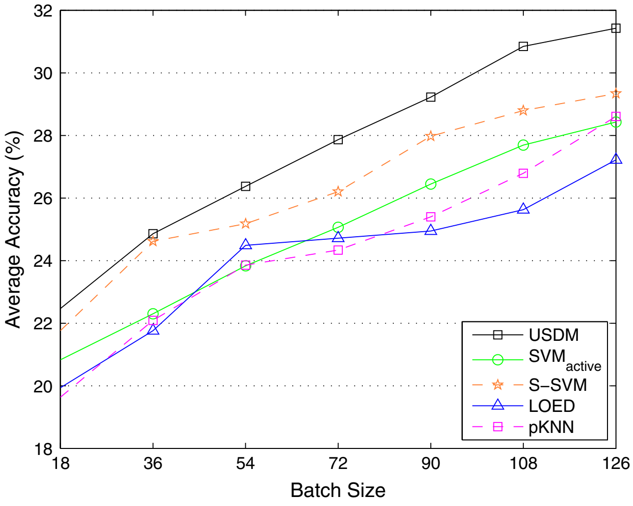


Fig. 6 A comparison of different active learning algorithms on complex event detection. There are 18 different events in this dataset

Figure 6 shows the experimental results on complex event detection. We can see that our algorithm consistently outperforms all the competitors. As shown in the figure, the advantage of our algorithm over other algorithms is quite visible. As the batch size grows, the performance of pKNN and S-SVM improves but is still worse than our algorithm. This experiment demonstrates that the algorithm proposed in this paper is more robust in dealing with "wild" data, compared to the state of the art.

5.6 Performance Comparison Using Different Seed Size

In this subsection, we examine the impact of the initial training size given that it usually plays a key role in the semi-supervised learning tasks. We perform this experiment by varying the seed size from 1 × c and 5 × c. As LOED is an unsupervised method that is irrelevant to the labelled seed set, we leave it out in this experiment. Figures 7, 8, 9, 10 and 11 show the experimental results on different datasets.

The experimental results in the figures (i.e., those in Fig. 7, 8, 9, 10 and 11) and the results when seed size is 3 × c demonstrate that for different seed sizes, our method consistently yields compelling performance, validating its efficacy in selecting the most informative data for a variety of vision tasks. Meanwhile, we notice S-SVM also obtains good performance, which further indicates that leveraging the unlabelled pool data does help improve the active learning performance.

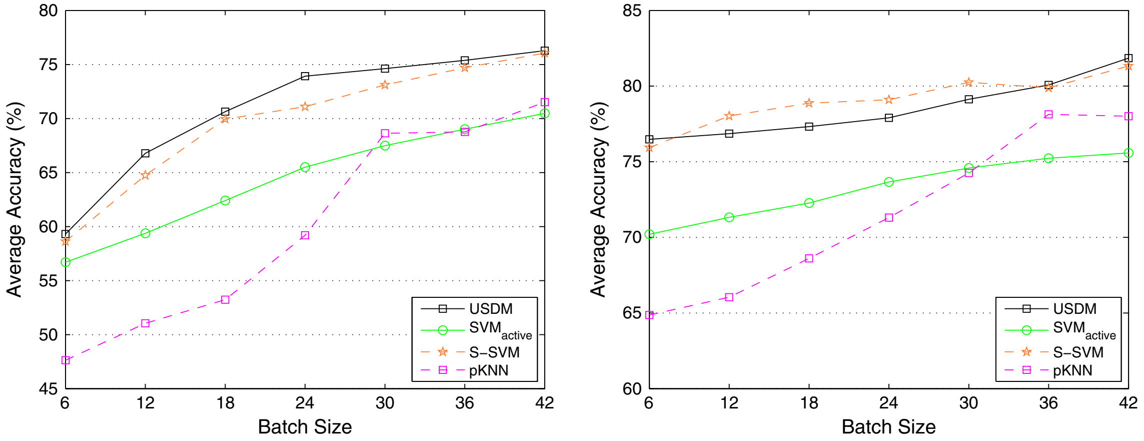
5.7 Performance Comparison on the Pool Data

From this subsection on, taking the Youtube action dataset as an example, we report the experimental results to test more characteristics of the proposed algorithm. These experiments include (1) classification accuracy of the active pool data; (2) classification accuracy when different classifiers are used; (3) classification accuracy when a different feature is used; (4) classification accuracy when the unweighted graph is used.

First, we evaluate the classification accuracy of different active learning algorithms on the pool data. To this end, we exclude the seed data and the selected batch data whereas treat the remaining pool data as the testing data. Since LOED is unsupervised, we leave it out in the comparison. Figure 12 displays the experimental results. We can see that our method consistently gains the top performance, whereas S-SVM and pKNN obtain good performance as well. This observation is consistent with the results when the testing data are outside the active pool, again demonstrating the effectiveness of the proposed USDM algorithm.

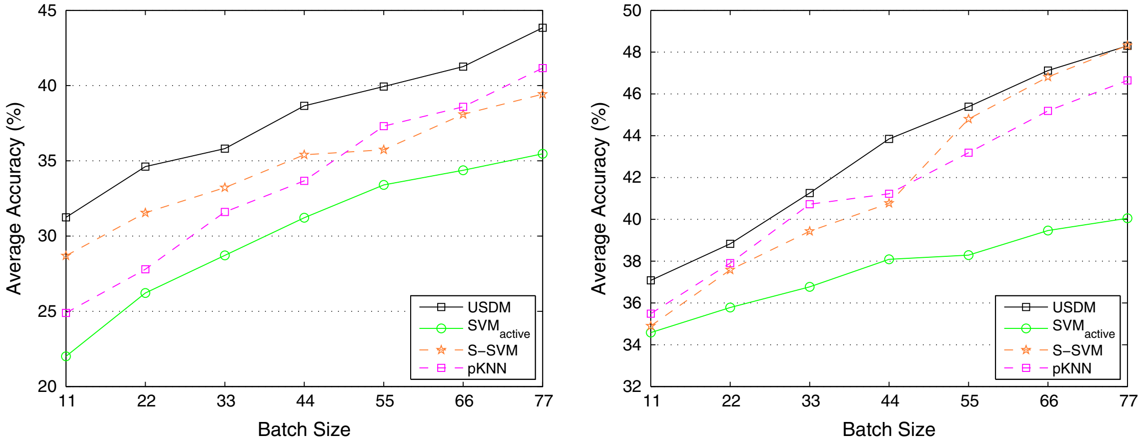
5.8 Performance Comparison Using a Different Feature

In this subsection, we compare the performance of the active learning algorithms when a different feature is used. In the



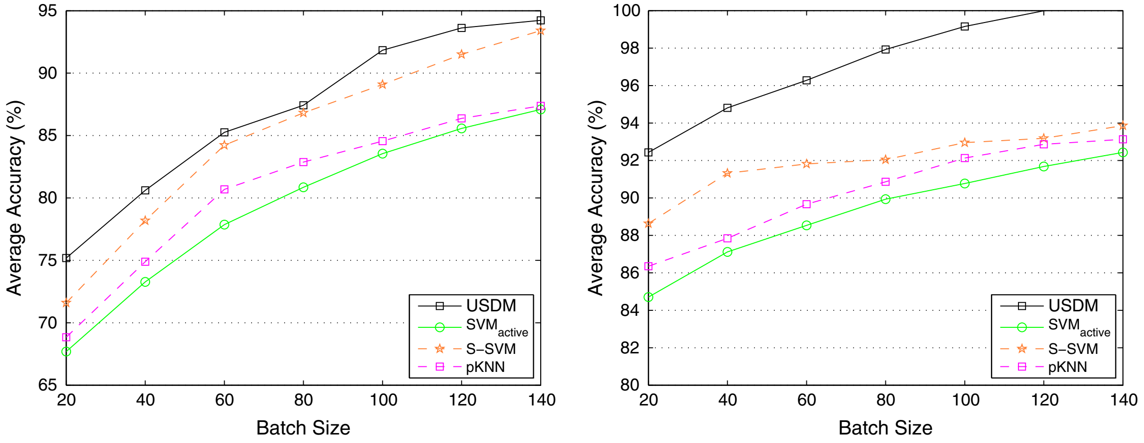
(a) seed size: c **(b)** seed size: $5 \times c$

Fig. 7 Performance comparison on KTH dataset *w.r.t.* different seed size. Our method is consistently competitive



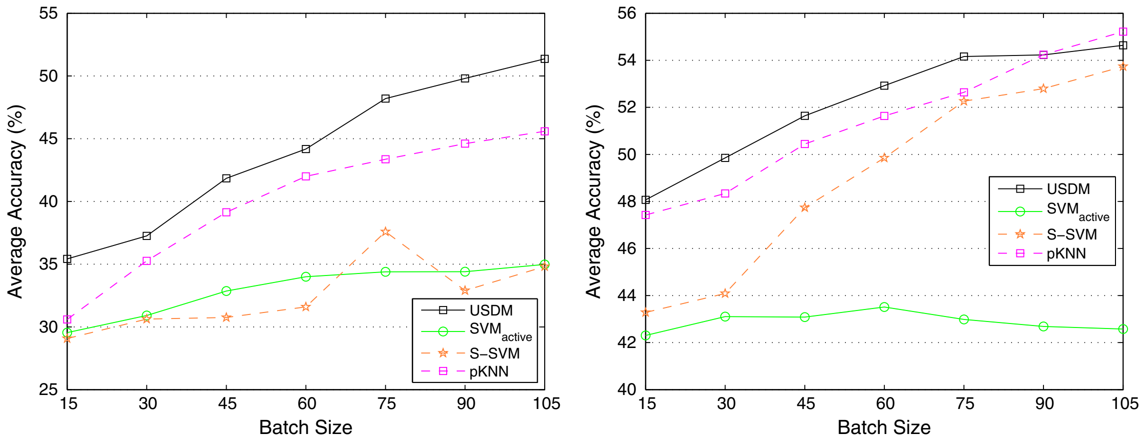
(a) seed size: c **(b)** seed size: $5 \times c$

Fig. 8 Performance comparison on Youtube dataset *w.r.t.* different seed size. Our method is consistently competitive



(a) seed size: c **(b)** seed size $5 \times c$

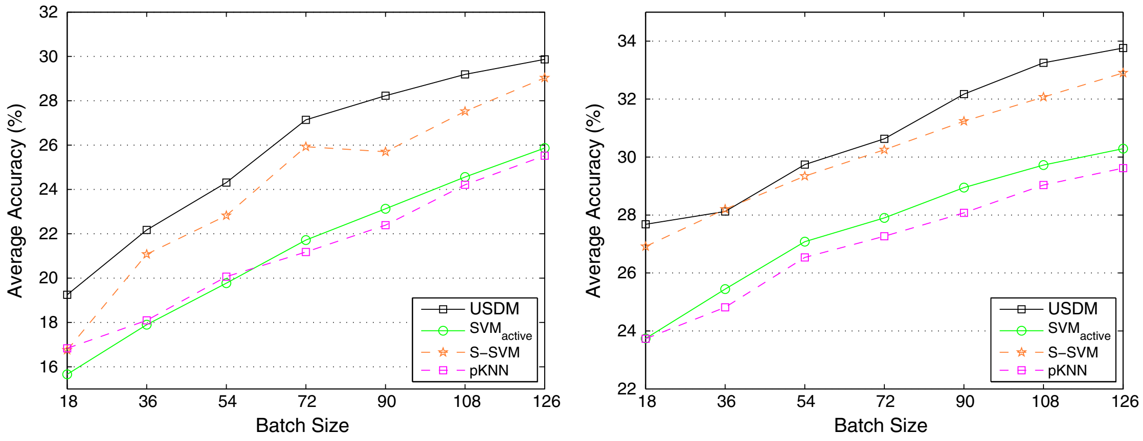
Fig. 9 Performance comparison on Coil dataset *w.r.t.* different seed size. Our method is consistently competitive



(a) seed size: c

(b) seed size: $5 \times c$

Fig. 10 Performance comparison on Scene15 dataset *w.r.t.* different seed size. Our method is consistently competitive



(a) seed size: c

(b) seed size: $5 \times c$

Fig. 11 Performance comparison on MED dataset *w.r.t.* different seed size. Our method is consistently competitive

previous experiments, we have used the STIP feature for action recognition. In this experiment, we use the MoSIFT feature (Chen and Hauptmann 2009) for action recognition. Figure 13 shows the experimental result, where a 1,000 dimension BoW MoSIFT feature is used to represent the videos in the Youtube dataset.

Comparing Figs. 2 and 13, we can see that the MoSIFT feature performs better than STIP feature on Youtube dataset. Similarly, our algorithm outperforms the other compared algorithms dramatically. In particular, when 11, 22, and 33 data are selected, our algorithm outperforms the second best algorithm by about 10 %, relatively. This experiment demonstrates that when a better feature is used, the performance of an active learning algorithm usually improves. Nevertheless, our algorithm outperforms the other competitors consistently using a different feature.

5.9 Performance Comparison Using Different Classifiers

The function of active learning algorithms is to select the most informative data for supervision and then use these labelled data as input of a specific classification algorithm to train a classifier for recognition. It turns out an interesting question how the active learning algorithms will perform if we use a different classifier. In this subsection, we again use Youtube dataset as a showcase to compare different active learning algorithms using some other classifiers.

We first use the Least Square Regression (LSR) as the classifier for action recognition. This time, LSR is used for all the active learning algorithms, including our USDM, SVM$_{active}$, S-SVM, LOED and pKNN. Each active learning algorithm is first performed to select the training data, based on which a LSR classifier is trained for action recognition. Figures 14

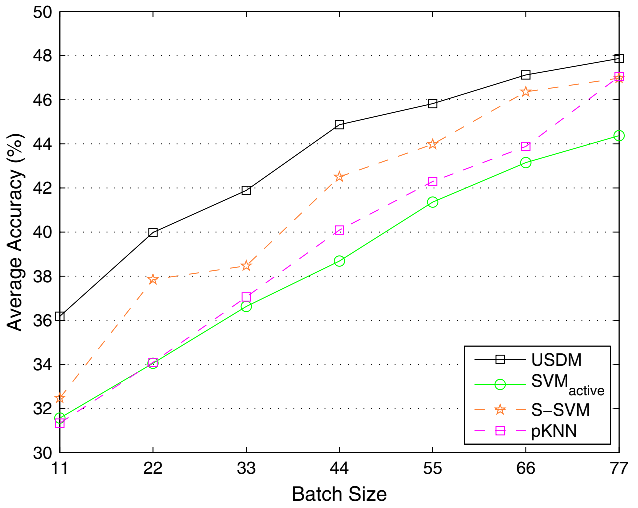


Fig. 12 A comparison of different active learning algorithms on classifying the pool data. The results are based on Youtube dataset

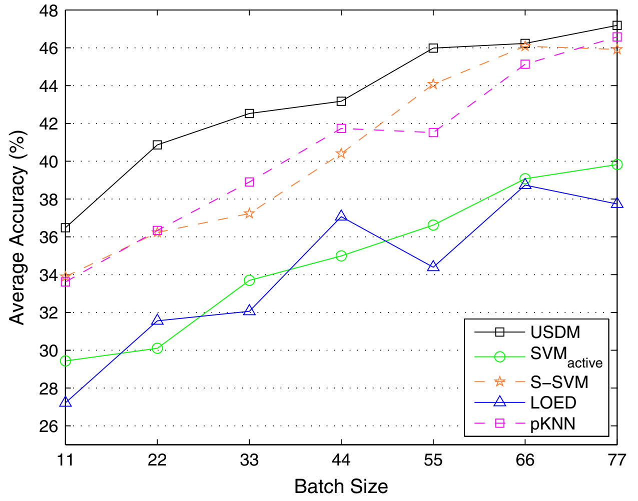


Fig. 13 A comparison of different active learning algorithms on action recognition using the MoSIFT feature. The results are based on the Youtube dataset

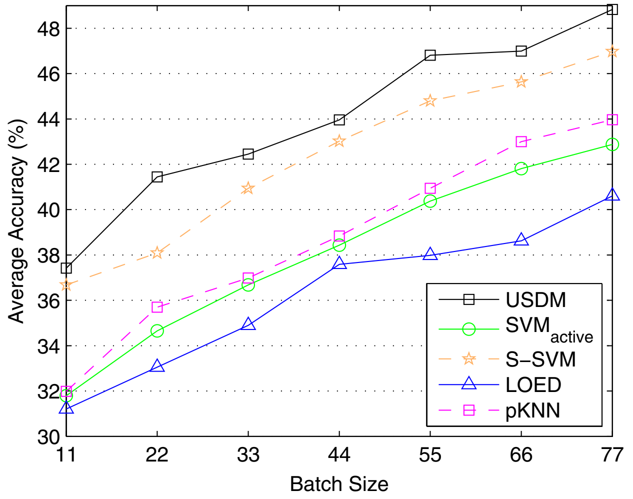


Fig. 14 A comparison of different active learning algorithms on Youtube dataset using STIP feature. In this experiment, least squares regression (LSR) is used as the classifier for action recognition

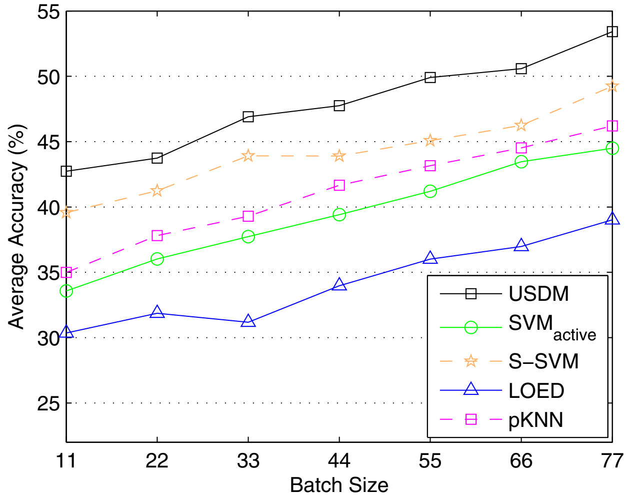


Fig. 15 A comparison of different active learning algorithms on Youtube dataset using MoSIFT feature. In this experiment, LSR is used as the classifier for action recognition

and 15 show the experimental results when STIP feature and MoSIFT feature are used, respectively. We can see from the two figures that the proposed algorithm USDM outperforms all the other algorithms at all batch sizes, when LSR is used as the classifier. This experiment further demonstrates that our algorithm is more effective than other active learning algorithms when a different classifier is used.

Next, we additionally use KNN as the classifier to compare the performance of different active learning algorithms on Youtube dataset. In this experiment, we use the same setting as LSR. Figures 16 and 17 show the experiment results when STIP feature and MoSIFT feature are used, respectively. For both features, when using KNN as the classifier, our algorithm dramatically outperforms the competitors. The visual concept recognition accuracy generally relies on three factors. The first one is the feature; the second one is the classifier and the third one is the data selected for supervision. We observe in the experiment that our USDM algorithm consistently outperforms the other methods when a different feature and/or a different classifier are used.

The experiment result of adopting different classifiers reported in this section also shows that the performance of our algorithm is still better than other compared algorithms when a different classifier is used. We observe similar performance on all datasets if an SVM classifier is directly trained instead of reconstructing a larger graph for random walk. Thus in real world applications one may directly train an inductive classifier, e.g., SVM, based on the data

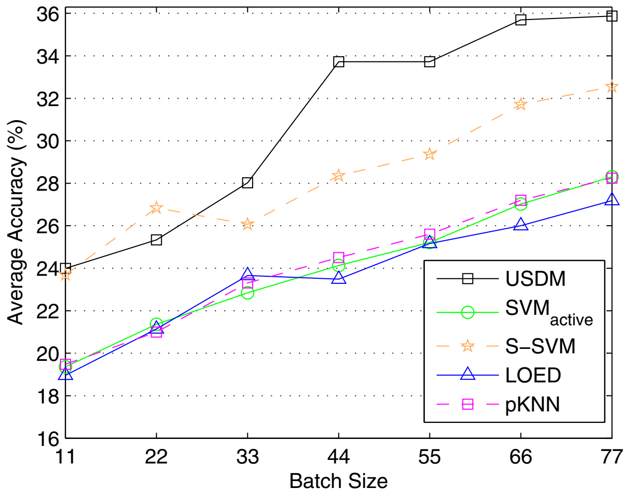


Fig. 16 A comparison of different active learning algorithms on Youtube dataset using STIP feature. In this experiment, KNN is used as the classifier for action recognition

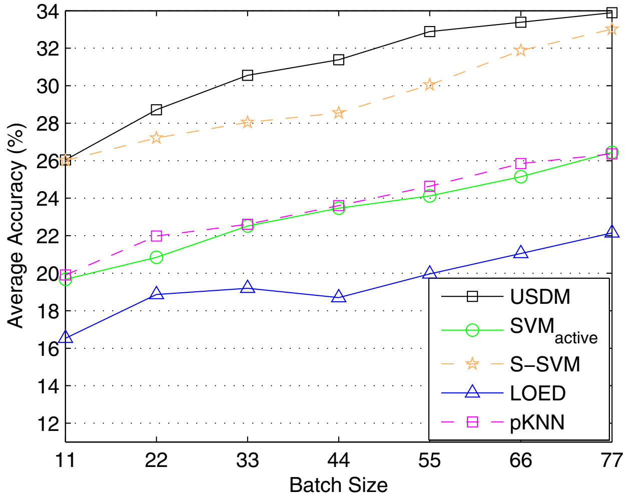


Fig. 17 A comparison of different active learning algorithms on Youtube dataset using MoSIFT feature. In this experiment, KNN is used as the classifier for action recognition

selected by the USDM algorithm to reduce the computation cost.

5.10 Performance Variation Using an Unweighted Graph

In the previous experiment, we use the weighted graph defined in (1) for the random walks. In the following experiment we compare the weighted graph with different parameter σ and the unweighted graph defined in (2). Again, we use the Youtube dataset to showcase with a randomly generated seed set of 33 videos. Figure 18 shows the experimental results.

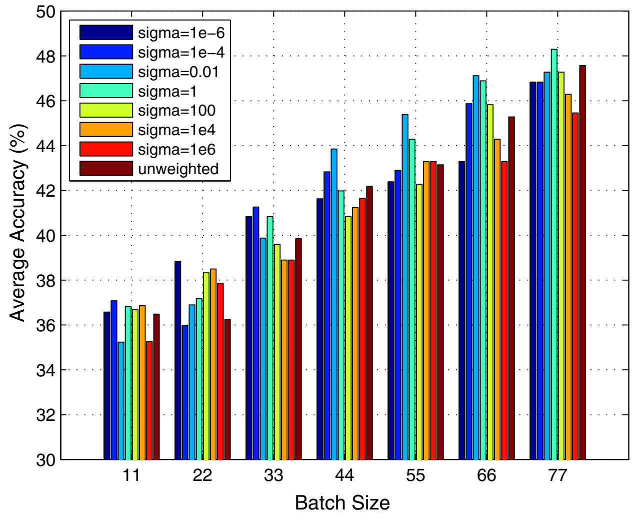


Fig. 18 Performance comparison between unweighted graph and weighted graph with different σ on Youtube dataset

It can be seen that if σ is appropriately chosen, the weighted graph usually gives us better performance. In other words, better performance can be expected when σ is optimal. In this experiment, we can see that the performance is usually better when σ is small. However, the optimal σ is data dependent, and can be determined by cross validation or experiments.

5.11 Computational Efficiency Comparison

Finally, taking MED dataset as an example, we compare the computational efficiency of the supervised and semi-supervised active learning algorithms. The computation time of the semi-supervised active learning algorithms mainly depends on the size of the active pool. In this experiment, the active pool size varies from 250 to 1,250, with an interval of 250. All experiments are implemented by Matlab R2011a, which is installed on a machine with 24 cores[2] and 64.0GB RAM.

Figure 19 shows the average time elapsed to select 18 data for labelling, i.e., the batch size is 1 × c. Note that only S-SVM, and our USDM exploit data distribution while SVM$_{active}$ and pKNN merely utilize the seed set for active learning. Thus the size of active pool does not affect speed much for pKNN and SVM$_{active}$. Although SVM$_{active}$ and pKNN are faster, their performance is worse in the previous experiments than the semi-supervised active learning algorithms S-SVM and USDM. In our experiments, as a semi-supervised active learning algorithm, S-SVM generally achieves the second best performance in accuracy. As shown in Fig. 19, our algorithm outperforms S-SVM dramatically in

[2] Intel(R)Xeon Processor, 24 cores

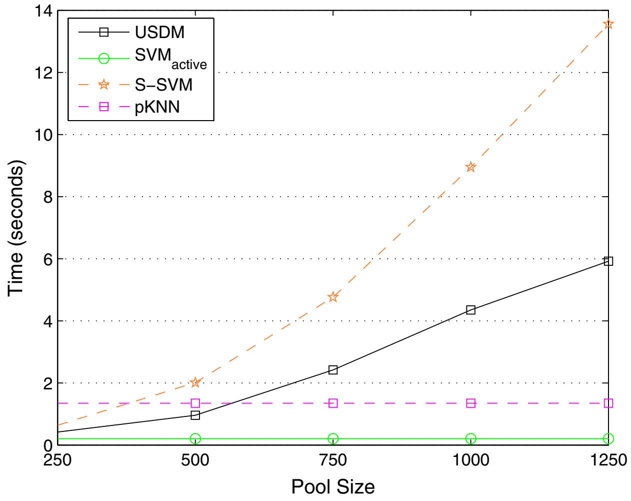


Fig. 19 Running time of different active learning algorithms *w.r.t.* different pool sizes. The result shown in this figure is the elapsed time (seconds) of selecting pool data for training

efficiency. If we increase the pool size, the efficiency advantage of our algorithm over S-SVM will become more visible.

6 Conclusion

Generally speaking, there are three important factors in visual concept recognition, which are the features, the classifiers and the data selected for supervision. In this paper, we have proposed a new active learning algorithm USDM for visual concept recognition. To address the problem of small seed set size in uncertainty sampling, we proposed to exploit the distribution of all the data in the active pool and the seed set. Considering that the uncertain data in the active pool are potentially similar to each other, we proposed to make the selection as diverse as possible. USDM is able to evaluate the "informativeness" of a sample across multiple classes, making the selection more accurate. An efficient algorithm was used to optimize the objective function of USDM. Extensive experiments on a variety of applications with different classifiers and features demonstrate that USDM dramatically outperforms the state of the art. We have observed that even if the size of the seed and pool sets is the same, the classification performance would be different when the the seed and pool sets are different. In our future research, we will study to optimally initialize the seed set and active pool.

Acknowledgments This paper was partially supported by the US Department of Defense the U. S. Army Research Office (W911NF-13-1-0277), partially supported by the ARC DECRA project DE130101311, and partially supported by the Tianjin Key Laboratory of Cognitive Computing and Application.

References

Bertsekas, D. (1999). *Nonlinear programming* (2nd ed.). Belmont, MA: Athena Scientific.

Brinker, K. (2003). Incorporating diversity in active learning with support vector machines. In *International conference on machine learning*.

Campbell, C., Cristianini, N., & Smola, A. J. (2000). Query learning with large margin classifiers. In *ICML*.

Chattopadhyay, R., Wang, Z., Fan, W., Davidson, I., Panchanathan, S., & Ye, J. (2012). Batch mode active sampling based on marginal probability distribution matching. In *KDD* (pp. 741–749).

Chen, M., & Hauptmann, A. (2009). Mosift: Recognizing human actions in surveillance videos. In Technical Report CMU-CS-09-161.

Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research (JAIR)*, *4*, 129–145.

Delbos, F., & Gilbert, J. (2005). Global linear convergence of an augmented lagrangian algorithm to solve convex quadratic optimization problems. *Journal of Convex Analysis*, *12*(1), 45–69.

Doyle, P. G., & Shell, J. (1984). *Random walks and electric networks*. Washington, DC: Mathematical Association of America.

Gong, B., Grauman, K., & Sha, F. (2014). Learning kernels for unsupervised domain adaptation with applications to visual object recognition. *International Journal of Computer Vision*, *109*(1–2), 3–27.

Han, Y., Yang, Y., Yan, Y., Ma, Z., Sebe, N., & Zhou, X. (2014). Semi-supervised feature selection via spline regression for video semantic recognition. *IEEE Transactions on Neural Networks and Learning Systems*. doi:10.1109/TNNLS.2014.2314123.

He, X., Min, W., Cai, D., & Zhou, K. (2007). Laplacian optimal design for image retrieval. In *SIGIR*.

Hoi, S., Jin, R., Zhu, J., & Lyu, M. (2008). Semi-supervised SVM batch mode active learning for image retrieval. In *CVPR*.

Hoi, S., Jin, R., Zhu, J., & Lyu, M. (2009). Semisupervised svm batch mode active learning with applications to image retrieval. *ACM Transactions on Information Systems*, *27*(3), 16:1–16:29.

Hoi, S., & Lyu, M. (2005). A semi-supervised active learning framework for image retrieval. *CVPR*, *2*, 302–309.

Jain, P., & Kapoor, A. (2009). Active learning for large multi-class problems. In *CVPR*.

Jegelka, S., Kapoor, A., & Horvitz, E. (2014). An interactive approach to solving correspondence problems. *International Journal of Computer Vision*, *108*(1–2), 49–58.

Joshi, A., Porikli, F., & Papanikolopoulos, N. (2009). Multi-class active learning for image classification. In *CVPR*.

Kapoor, A., Grauman, K., Urtasun, R., & Darrell, T. (2010). Gaussian processes for object categorization. *International Journal of Computer Vision*, *88*(2), 169–188.

Kowdle, A., Chang, Y., Gallagher, A., & Chen, T. (2011). Active learning for piecewise planar 3D reconstruction. In *CVPR*.

Laptev, I., Marszalek, M., Schmid, C., & Rozenfeld, B. (2008). Recognizing realistic actions from videos in the wild. In *CVPR*.

Lazebnik, S., Schmid, C., & Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*.

Li, H., Shi, Y., Chen, M., Hauptmann, A., & Xiong, Z. (2010). Hybrid active learning for cross-domain video concept detection. In *ACM Multimedia*.

Li, M., & Sethi, I. K. (2006). Confidence-based active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(8), 1251–1261.

Li, X., Wang, L., & Sung, E. (2004). Multilabel SVM active learning for image classification. In *ICIP*.

Lindenbaum, M., Markovitch, S., & Rusakov, D. (2004). Selective sampling for nearest neighbor classifiers. *Machine Learning*, *54*(2), 125–152.

Liu, J., Luo, J., & Shah, M. (2009). Recognizing realistic actions from videos in the wild. In *CVPR*.

Ma, Z., Yang, Y., Nie, F., Sebe, N., Yan, S., & Hauptmann, A. (2014). Harnessing lab knowledge for real-world action recognition. *International Journal of Computer Vision*, *109*(1–2), 60–73.

Ma, Z., Yang, Y., Sebe, N., & Hauptmann, A. (2014). Knowledge adaptation with partiallyshared features for event detection using few exemplars. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(9), 1789–1802.

Nene, S., Nayar, S., & Murase, H. (1996). Columbia object image library (coil-20). Technical Report CUCS-005-96.

Schüldt, C., Laptev, I., & Caputo, B. (2004). Recognizing human actions: A local SVM approach. In *ICPR*.

Shen, H., Yu, S.-I., Yang, Y., Meng, D., & Hauptmann, A. (2014). Unsupervised video adaptation for parsing human motion. In *ECCV*.

Spielman, D., & Teng, S.-H. (2004). Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *STOC*.

Tenenbaum, J., Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*(5500), 2319–2323.

Tong, S., & Chang, E. (2001). Support vector machine active learning for image retrieval. In *ACM Multimedia*.

Vondrick, C., & Ramanan, D. (2011). Video annotation and tracking with active learning. In *NIPS*.

Wang, L., Chan, K. L., & Zhang, Z. (2003). Bootstrapping SVM active learning by incorporating unlabelled images for image retrieval. In *CVPR* (pp. 629–634).

Yan, R., Yang, J., & Hauptmann, A. (2003). Automatically labeling video data using multi-class active learning. In *ICCV*.

Yang, Y., Ma, Z., Hauptmann, A., & Sebe, N. (2013). Feature selection for multimedia analysis by sharing information among multiple tasks. *IEEE Transactions on Multimedia*, *15*(3), 661–669.

Yang, Y., Ma, Z., Xu, Z., Yan, S., & Hauptmann, A. (2013). How related exemplars help complex event detection in web videos. In *ICCV*.

Yang, Y., Nie, F., Xu, D., Luo, J., Zhuang, Y., & Pan, Y. (2012). A multimedia retrieval framework based on semi-supervised ranking and relevance feedback. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(4), 723–742.

Yu, K., Bi, J., & Tresp, V. (2006). Active learning via transductive experimental design. In *ICML* (pp. 1081–1088).

Zhu, X. (2008). Semi-supervised learning literature survey. Technical Report, University of Wisconsin-Madison.

Zhu, X., Ghahramani, Z., & Lafferty, J.D. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In *ICML* (pp. 912–919).