Robust Real-Time Tracking with Diverse Ensembles and Random Projections

Ahmed Salaheldin
Center for Informatics Science,
Nile University
Giza, Egypt
ahmed.hussein@nileu.edu.eg

Sara Maher
Center for Informatics Science,
Nile University
Giza, Egypt

Mohamed El Helw
Center for Informatics Science,
Nile University
Giza, Egypt

Abstract

Tracking by detection techniques have recently been gaining popularity and showing promising results. They use samples classified in previous frames to detect an object in a new frame. However, because they rely on self updating, such techniques are prone to object drift. Multiple classifier systems can be used to improve the detection over that of a single classifier. However, such techniques can be slow as they combine information from different tracking methods. In this paper we propose a novel real-time ensemble approach to tracking by detection. We create a diverse ensemble using random projections to select strong and diverse sets of compressed features. We show that our proposed ensemble tracker significantly improves the accuracy of tracking while not using any additional information than that available to the single classifier; thus requiring little extra computational overhead. Our results also show that employing our multiple classifier system with feature subsets gives significantly better results than directly combining the features.

1. Introduction

We deal with the problem of object tracking from a video stream. Object tracking is an important task in many applications such as robotics, surveillance, augmented reality and traffic sensing. However, creating a generic object tracker is challenging due to the restrictions on time and computations, and the lack of information about the object and its surroundings. In recent years tracking by detection has gained wide popularity in the literature and has yielded promising results [1][2][3][4][5]. Tracking by detection treats the tracking problem as a binary classification task. In each frame a classifier is used to classify patches of the scene as object or background.

Tracking by detection can be performed in an offline or online manner. In this paper we are concerned with the latter. Offline detection requires training a supervised learner beforehand, which will then be used to detect the object throughout the sequence. In online detection, the tracker is automatically updated in a semi-supervised manner using information from previous frames. Online tracking is useful for adapting to changes in the object or the background. Moreover, it may be necessary if there are not enough labeled examples or time to train a supervised classifier in advance. However, since this approach is self learning (i.e. uses its own predictions for online training), tracking by detection is prone to update errors. False positives used to update the classifier cause the detection to degrade. Moreover, self updating with slightly inaccurate detections over time can lead to the tracker drifting away from the object into the background.

Multiple Classifier Systems (MCS) are a machine learning paradigm based on building ensembles of classifiers to improve upon the performance of a single classifier. MCS have been used in tracking by detection applications to improve the accuracy and robustness of tracking. For example, Boosting can be used to add a number of classifiers to form a strong tracking ensemble [5][6][7][8]. Visentini et al [9] propose a more flexible approach than boosting that constructs an ensemble by promoting pairwise disagreement between individual classifiers. Other approaches include combining multiple learning algorithms or feature sets to exploit different information. We will review some of these techniques in section 2.

A key condition to creating a successful classifier ensemble is diversity. Diversity, which entails a degree of disagreement among different classifiers, provides wider knowledge and is more robust due to tolerance to errors of single classifier drift. If the classifiers were similar, there would be no gained information from combining them. Minku et al [10] show that diversity in online learning algorithms reduces the error when concept drift occurs, and that high diversity is necessary to deal with severe drifts.

Moreover by assigning classifiers with different areas of expertise, classifier ensembles can be accurate in a variety of situations and robust to dramatic changes in the object or the environment. For example, a common problem in tracking is adapting to changes in the object. If the model is rapidly adaptive, it may adopt background information and so can easily drift. If the model is rigid, the object may no longer be recognized when concept drift (change in the object) occurs. Using different experts with varying degrees of rigidity can alleviate this problem. Nishida et al [11] use an ensemble of online and offline classifiers, and a drift detection mechanism to deal with concept drift and recurring concepts. The same idea can be applied to problems where the environment changes rather than the object (such as changes in lighting) rather than using one model to fit all situations.

In this paper we propose a novel approach for real-time robust object tracking from video streams. We aim to use multiple classifier systems to perform tracking by detection that is robust against drift and changes in the object. Random projections provide a strong but sparse representation of the original signal and so are suitable for use in ensemble methods. By using Random, sparse low dimensional projections we can build a diverse ensemble while maintaining real-time performance. We show that our method of creating ensembles satisfies the conditions of having strong and diverse classifiers, and is adequate for use in a semi-supervised online problem. We investigate infusing further diversity to the ensemble by changing the speed with which the individual classifiers adapt to concept drift. Fusion methods are also investigated to maximize the tolerance of the final output for individual errors. Quantitative results show that the proposed ensemble tracker significantly improves the accuracy of single classifier tracking. The results also show that employing our multiple classifier system with feature subsets gives significantly better results than directly combining the features.

The paper is organized as follows: in the next section we survey related work. Section 3 explains preliminary concepts and techniques where we explain the theoretical hypothesis behind our contribution. Section 4 describes the proposed method. In Section 5, we present our experiments and obtained results compared to single classifiers and the case when features are directly combined. Section 6 concludes the paper.

2. Related Work

In this section we survey related work, illustrating the benefits of employing multiple classifier systems in tracking and the necessity of diversity in such systems. We also outline the challenges of integrating several models for real-time robust tracking.

2.1.1 Multi-Cue Tracking

One form of ensembles used in object tracking is multi-cue tracking. In this case, multiple feature sets such as intensity changes and color maps are extracted and used to train the constituents of the ensemble. Triesch et al [12] use 5 visual cues to construct ensemble of trackers. The decisions of the classifiers are fused using Democratic Integration (DI) [13] where all cues agree upon common position estimation; and then use this estimation to adapt the parameters of the individual cues and redistribute their weights in the ensemble. In order for democratic voting to yield robust decisions, it is assumed that environmental changes in the scene only affect a minority of the ensemble. Wu et al [14] also use multiple visual cues providing shape and color representations of the object. By integrating a number of rough visual cues using co-inference they achieve a rich representation of the object based on the hypothesis that the inference in a higher dimensional state space can be factorized by several lower dimensional subspaces in an iterative fashion.

Spengler et al [15] use a multi cue ensemble to achieve more robust tracking. They utilized information from different sources to train the classifiers constituting the ensemble. Each classifier trained by a different cue proposes its independent estimation of the target. They try both Democratic Integration and Condensation [16] as fusion strategies. They report that Democratic Integration is sensitive to false positives. Upon the tracking of false positives the internal parameters will adjust towards this false model. Due to the self-adaptive nature of the system, it will continue to accumulate error and not recover.

Kwon et al [17] use both visual cues and motion cues to create a tracking model. Moreover, they employ different template models for object representation. A set of observation models (or visual cues) is obtained by applying various feature extraction methods. Sparse principal component analysis (SPCA) is used to reduce the dimensionality of the feature sets and encourage diversity among the visual template models. Similarly, different motion models are employed to deal with smooth and abrupt motion of the object. The observation and motion models are combined to produce a number of basic trackers which are fused using an interactive Markov chain Monte Carlo framework to produce the final tracker.

In general, it can be noted that multi cue approaches produces robust results due to incorporating different information from varying sources. However, it may be time consuming due to the need to extract several feature groups. For example Kwon et al report that it takes 1-5 seconds per frame to run the tracker [17].

2.1.2 Multiple Method Tracking

Stalder et al [18] divide the problem into three tasks, namely detection, recognition and tracking. Each task is assigned to a different classifier. An offline classifier is used to reliably find the object of interest. A supervised online classifier provides trustworthy recognition. Updates are performed conservatively. Finally the tracker employs a semi-supervised online classifier that compromises reliability to be able to adapt to concept drift. Santner et al [19] use trackers with varying degrees of adaptability to cope with concept drift while maintaining robustness. Template matching (NCC) is used as a static tracker to detect the sample closest to the original object. Online Random Forests (ORF) act as a moderately adaptive appearance based tracker. A mean shift optical flow tracker (FLOW) represents the far end of the adaptability scale to adapt to fast appearance changes. The three trackers are fused by a rule engine: ORF overrules FLOW if it has confidence above a given threshold. ORF is updated only if it overlaps with NCC or FLOW to avoid wrong updates.

Siebel et al [20] combine multiple tracking hypotheses to track people more robustly. A motion detector is used to separate moving objects from the background, and then three models cooperate to produce the final tracking hypothesis: A region detector for tracking all moving regions detected by the motion detector. A head detector looks for heads in the detected regions and helps to initialize and validate tracking for a person. Finally, an active shape tracker detects and tracks the outline of a person.

As with multi-cue tracking, using multiple algorithms can be computationally intensive as it usually needs to linearly combine the run-time costs of several algorithms. To alleviate this problem, Santner et al [19] implement their method to run in parallel on multi core GPUs to achieve real time performance.

Contrary to related work, the proposed approach creates a diverse ensemble without utilizing multiple tracking algorithms or feature extraction techniques. Diverse classifier ensembles are created using only the information provided for a single tracker, which saves computational time and allows for the possibility of further combining the tracker with other approaches.

3. Preliminaries

3.1. Diversity

Diversity is an important requirement for creating successful ensembles since if all the constituents of an ensemble are similar then there would be no added benefit from combining them. Nevertheless, there is no unified method for measuring diversity, or even a unified definition for that matter [21]. Some authors try to establish, through theoretical analysis or heuristics, the importance of diversity in ensembles. For instance, Hansen and Salamon [22] show that diversity and accuracy of the individual classifiers is essential for a majority voting classifier ensemble. By using the binomial theorem [], given that all classifiers have a probability of correctly classifying an instance $(1-p)$, and assuming that their errors are statistically independent, the probability of the ensemble incorrectly classifying and instance is given by:

$$P(ensemble\ incorrect) = \sum_{k>(M/2)}^{M} \binom{M}{k} p^k (1-p)^{(M-k)} \quad (1)$$

where M is the total number of classifiers, k is number of classifiers with incorrect predictions about the sample and p is the probability of a wrong prediction by the individual classifiers. Using majority voting as a fusion rule, more than half the ensemble must be wrong in order for the ensembles decision to be wrong. Assuming the individual classifiers perform better than random guess, the probability for ensemble misclassifications given by Eq. (1) is very low for sufficiently large ensembles. As the number of classifiers in the ensemble increases, the probability for error becomes smaller. This calculation of the probability of misclassification is made under the condition that the classifiers are statistically independent. Such a condition may not be feasible in real applications; nevertheless, Eq. (1) gives theoretical backing to the intuition that diverse predictors combined by voting will improve upon the performance of single predictors.

In the context of tracking, Li et al [23] make similar assumptions and show that the error of a directly combined ensemble of classifiers tends to 0 as the size of the ensemble grows. Given an ensemble where the probability maps of the predictors are linearly combined,

$$\bar{q}^{t+1}(x) = \frac{1}{n}\sum_{i=1}^{n} q_i^{t+1}(x) \quad (2)$$

the decision of the ensemble for sample *x* is calculated as:

where n is the number of classifiers, $q_i^{t+1}(x) \in [0,1]$ is the class prediction of classifier i for sample x at frame $t+1$ and $\bar{q}^{t+1}(x)$ is the class output of the ensemble for sample x at frame $t+1$. Lemma 1 shows that $\bar{q}^{t+1}(x)$ can converge to the ground truth $p^{t+1}(x)$ exponentially [23].

Lemma 1. *Taking the average of the predictions from all classifiers as calculated in Eq(2), the average is bounded in a PAC sense. Assuming that the n trackers are independent and unbiased, then* $\bar{q}^{t+1}(x) \rightarrow p^{t+1}(x)$ *as* $n \rightarrow +\infty$

Proof. For any small $\epsilon > 0$ with Hoeffding inequality , we get that $P(|\bar{q}^{t+1}(x) - p^{t+1}(x)| \geq \epsilon) \leq 2\exp(-2n\epsilon^2)$.

However linearly combining the outputs of the classifiers has a large draw back in that a single wrong prediction by a classifier can sway the whole ensemble to the wrong decision if it has high confidence compared to the rest of the ensemble [23].

Studies of diversity extend to multiple classifier systems in semi-supervised learning. Co-training is a semi-supervised learning algorithm that uses two views of the data to teach each other in an iterative online manner [24]. As an example in a web classification task, features extracted from body of the page and features extracted from links pointing to the page serve as different cues. Blum and Mitchell [24] put forth two requirements for successful co-training; firstly that each view should be sufficient for classification, and secondly that the two views are independent given the class. The requirement for independence of the views has been relaxed by Balcan et al [25], but nonetheless maintained conditions for disagreement between the views in order for the respective features to hold complementary information. We will further discuss co-training and the ability of random feature subspaces to satisfy the diversity conditions for multi view semi supervised learning in the next segment.

3.2. Random Subspace Method

Random Subspace Method (RSM) is a technique to create lower dimensional feature subsets using random selection [26]. It was originally designed to split high dimensional features among an ensemble of decision trees. However, it can be used for feature selection and dimensionality reduction in general [27][28]. Breiman [29] argues that random forests can be favorable to other ensemble creating techniques such as bagging and boosting as they are simpler and faster while maintaining similar or better accuracy and robustness. An upper bound for the generalization error of random forests given as [29]:

$$\mathrm{PE} \leq \bar{\rho}(1-\mathrm{s}^2)/s^2 \tag{3}$$

where $\bar{\rho}$ is the mean value of the pairwise correlation between the classifiers and s is the strength (or accuracy) of the ensemble. Although the bound is likely to be loose, the equation shows that the two ingredients involved in the generalization error for random forests are the strength of the individual classifiers, and the diversity of the ensemble; the higher the strength and the lower the correlation, the lower the error of the ensemble [29].

Random selection is also proven useful in online semi-supervised learning. When two sufficient cues are not available for co-training, random splitting is employed. Nigam and Ghani [30] showed that random splitting can be successful if there is sufficient redundancy in the original feature space. Moreover, Slivka et al [31] employ a random subspace method to create an ensemble of semi-supervised learners. They show that creating an ensemble of co-training classifiers each employing a different random split can improve the performance. Even though no new features are added but merely the same features are split randomly several times, the ensemble is diverse enough to improve the performance.

3.3. Random Projections

Random projections are a method of projecting a high dimensional space to a lower dimensional space while preserving most of the information in the original signal. The theory relies on the Johnson-Lindenstrauss lemma [32] that states that with high probability the distance between a pair of samples in a high dimensional space is preserved if they are projected onto a randomly selected subspace with sufficient dimensions [33]. A random matrix *R* satisfying the Johnson-Lindenstrauss lemma can project compressive signals such as images onto a lower dimensional space without losing most of the information [34]. This allows for the use of robust features in detection without the computational cost that comes with high dimensional feature spaces.

A random matrix *R* of size $m \times n$ projects data from an *m* dimensional space to an *n* dimensional space where *n* is much smaller than m. *R* is usually a Random Gaussian matrix with zero mean and unit variance. However, to further reduce the computational cost, a sparser random matrix is employed where a member of *R*, r_{ij} is given by the equation:

$$r_{ij} = \sqrt{s} \times \begin{cases} 1\; with\; probability\; \frac{1}{2s} \\ 0\; with\; probability\; 1 - \frac{1}{s} \\ -1\; with\; probability\; \frac{1}{2s} \end{cases} \tag{4}$$

This matrix satisfies the Johnson-Lindenstrauss lemma for *s*= 2 or 3 [32]. For *s*=2, half the matrix is zeros and therefore half the computations for the projections may be ignored. For *s*=3, two thirds of the matrix is zeros. This not only saves computational load but also memory as only the non-zero elements of the matrix *R* are stored.

4. Proposed Method

4.1. Ensemble

4.1.1 Classifier Diversity

We propose to create a diverse ensemble by using random projections to create lower dimensional representations of Haar-like features [35]. Haar-like features are often used in object detection; however, they occupy very high dimensions and are therefore computationally intensive. For a single random matrix to maintain the information from Haar-like features, the number of projected dimensions should satisfy the Johnson-Lindenstrauss lemma [32]:

$$n \geq \left(\frac{4 + 2\beta}{\frac{\epsilon^2}{2} - \epsilon^3/3} \right) \ln d \qquad (5)$$

Zhang et al [33] calculate the lower bound for *n* to be 1600 but they report that 50 gives good results for tracking.

We use multiple random matrices to project Haar-like features into lower dimensional features spaces. The key to this approach is the ability of the compressed features to maintain the information in the original signal in a sparse representation. If the projections do not represent the original signal, the classifiers cannot be sufficient for classification. On the other hand, if we use high dimensional projections, all the projected feature subsets will carry the same information. Such tradeoff between the accuracy and diversity is essential for successful classifier combination. For this reason, artificially infusing diversity in a classifier may result in poor performance due to increased individual error [36]. Random projections present good representations of the original signal. However, with a low dimensional projection, the classifier is very sensitive to the random matrix. The performance may vary greatly according to the matrix used (this is clear in the results section). This allows us to capitalize on the conditions of accuracy and diversity by employing Sparse random matrices using Eq (4) to project high dimensional features to low dimensional feature spaces (much lower than the lower bound of the Johnson-Lindenstrauss lemma). The accurate representation of the projections guarantees with high confidence the strength of the individual classifiers. Additionally the low dimensional and sparse nature of the projections provide features subsets that are diverse enough to hold different information.

To further make use of the multiple classifiers, the classifiers are updated at different rates. This way each classifier has different degree of adapting to change. This increases the diversity among the ensemble and enables it to robustly deal with concept drift.

4.1.2 Classifier Fusion

Classifier fusion in tracking is different than fusion in other classification tasks in that the ensemble needs to agree on a single position for the object. A majority vote only considers the best candidate from each classifier and neglects the classifier predictions on the rest of the selected samples. Owing to individual errors a consensus is hard to reach among the best candidates. It is useful then to utilize the classifiers' decisions for more samples in the voting process. Direct combination of classifier outputs compares the scores of all candidates and selects the sample with the highest average score. However, this approach is not robust to individual errors as it is sensitive to classifier confidences and one confident incorrect prediction can mislead the ensemble into a wrong decision [23].

To overcome these limitations we use a ranking based method to combine the classifiers. The 10 most confidently predicted samples are selected and assigned ranks based on the equation:

$$r_i = \sum_{j}^{n} (C - p_{ij}) \qquad (6)$$

where n is the number of classifiers, *C* is a constant and p_{ij} is the position of sample i in the top 10 list for classifier j. The lists are ordered in descending order of prediction confidence, so the most confident sample is assigned position 1 and consequently has the highest rank. If a sample i is not present in the list for classifier j then p_{ij} = *C* and so it is not assigned a rank by that classifier. The Constant *C* controls how much the position of *i* in the list contributes to the rank. For example $C = 11$means that a sample that is number 1 on the list of only a single classifier has a higher rank than a sample that is number 10 on the lists of 9 classifier. As *C* becomes larger, the number of classifiers that assign i a rank becomes more important than its position in the lists.

After assigning ranks, the sample i with the highest rank r_i is chosen as the new position of the object. This way our ranking based fusion is robust against confident but incorrect predictions. Even if the individual selections for all classifiers are inaccurate, a majority of the classifiers can still agree on a sample in their top 10 lists and assign it a higher rank than the top candidates. This tolerance to error encourages us to add diversity to the ensemble at the cost of robustness of individual classifiers such as varying update rates, where highly adaptable learners are very prone to drift.

4.2. Algorithm

We treat the tracking problem as a classification task. Given the bounding box for the object in the first frame, positive samples are gathered near the object and negative samples surrounding the object. These samples are used to train multiple classifiers each with features projected from a different random matrix. Patches are sampled around the previous position and each classifier produces a prediction for the detected samples. For each classifier the same random matrix used for the training samples is used for the detected samples. For classifiers we use the ratio classifier in [33]. The outputs of the classifiers are fused using Eq(6) to select the final output of the ensemble.

In frame $t + 1$ positive and negative samples are generated relative to the position chosen by the ensemble in frame t and are used to update the classifiers. The predictions of individual classifiers do not directly update the rest of the ensemble so there is no risk from the classifiers becoming more similar over time. The sample generation step which takes a lot of computational time is unified for all classifiers, and the different random matrices are generated once at the start of the algorithm, so using multiple classifiers adds little computational over head to the single classifier approach and runs in real time.

The pseudo code for our algorithm is shown below.

```
Algorithm
  Given: Bounding box $B$ at frame $t$
         Array of Random Matrices $R = \{R_1..R_n\}$
         Array of Ratio Classifiers $H = \{H_1..H_n\}$
  Do:
    ▪ Sample a positive sample set $P$ and a negative sample set $N$ with respect to $B$.
    ▪ Use R to generate $n$ Training sets $T_1..T_n$ from $P$ and $N$.
    ▪ Update classifiers $H_1..H_n$ using $T_1..T_n$ respectively.
    ▪ In frame $t+1$sample a set of Image patches $D$ around $B$.
    ▪ Extract feature sets from $D$ using $R$ and use each classifier in $H$ to perform predictions on the samples in $D$.
    ▪ Fuse the outputs of classifiers $H_1..H_n$ using Eq(6) to produce a rank $r_i$ for each sample $i$ of $D$

  Output: bounding box of sample $i$ with maximum rank $r_i$
```

5. Experiments

5.1. Experimental Setup

The proposed method is applied on a number of benchmark sequences and quantitative evaluation is presented. We compare the diverse ensemble approach to using a single classifier. A single classifier trained using random projections has been shown to compare favorably with state of the art tracking techniques in [33]. For a fair comparison, since the classifiers are sensitive to the random matrix used to extract the features, we use the same random matrices used by the single classifier for the ensemble. For each experiment, 100 random matrices are generated and the single-classifier tracker is run on the sequence 100 times each time using a different random matrix. The average performance of the runs is reported. For the ensemble approach, each classifier is assigned one of the 100 random matrices. The experiment is repeated until the random matrices run out and the average performance is reported. In our experiments we use ensembles of size 3, 5 and 10. We also compare our approach to a single classifier that uses 10 times the number of features used by each classifier in the ensemble. This is to evaluate the benefits of using multiple classifier systems over directly combining the features. Again the experiment is run 100 times and the average is reported.

For each sequence, the parameters are optimized on the single classifier and remain fixed for all experiments on the sequence. The parameters for the tracker are α: the radius for drawing positive samples from around the bounding box B, ζ *and* β: the inner and outer radius for drawing negative samples around B, γ the radius for sampling patches to detect the new position of the object, λ: the learning rate for updating the classifiers, where the lower the learning rate, the faster the classifier is updated. The parameters used for each sequence are provided in Table 1. The number of features n for each classifier is 50.

Sequence	α	ζ	β	γ	λ
David indoors	4	8	30	8	0.9
Lemming	4	120	150	70	0.9
Sylvester	4	8	30	20	0.85
Tiger2	4	8	30	20	0.9
Cow	4	120	150	70	0.9
Face	4	8	30	20	0.9
Twinings	4	8	30	20	0.9
Sunshade	4	8	30	20	0.9
Car	4	8	30	20	0.9

Table 1: tracker parameters for different sequences

The ensemble experiments are carried out one time with the fixed λ used by the single classifier and another time where the learning rate for classifier i is given as $lr_i = 0.5 + \frac{i}{n}(\lambda - 0.5)$. Where n is the number of classifiers. So the learning rates of the classifiers are uniformly distributed from 0.5 to λ. The experiments are also repeated once for C in Eq (6) = 11 to give higher weight to the individual classifiers confidence, and once with C =100 to give a higher weight to the consensuses of the ensemble. We quantify the performance on the sequences using a measure of center location error (CLE). In this case, the Euclidian distance in pixels between the center of the detected bounding box and the provided ground truth is calculated.

All experiments are conducted on an i7-920 2.66GHz machine running Matlab on Windows 7. The single classifier trackers with n=50 and n=500 run at ~32 and 14 frames per second (FPS) respectively. While the ensemble classifier trackers of size 3, 5 and 10 run at ~25, 20 and 14 fps respectively.

5.2. Results

Table 2 presents the results for our experiments using different approaches. The $Single_{50}$ column shows the average error (and standard deviation) for the single classifier with n=50. $Single_{500}$ shows the results of a single classifier with n=500. The letter E signifies an ensemble; the number following E indicates the size of the ensemble. The superscript shows the value of C and the subscripts S and D indicate if the learning rate is static (fixed for all classifiers) or dynamic (each classifier has a different learning rate) respectively. **Bold** numbers indicate the best results and *Italics* indicate the second best

Sequence	$Single_{50}$	$Single_{500}$	$E3_D^{100}$	$E3_D^{11}$	$E3_S^{100}$	$E3_S^{11}$	$E5_D^{100}$	$E5_D^{11}$	$E5_S^{100}$	$E5_S^{11}$	$E10_D^{100}$	$E10_D^{11}$	$E10_S^{100}$	$E10_S^{11}$
David	17.1	13.5	12.3	13.1	13.4	13.5	**9.5**	*10.7*	15.2	15.7	**9.5**	**9.4**	16.4	14.9
Lemming	70.6	58.2	46.2	50.5	58.3	55.3	44	*39.8*	53.8	59.9	**27.8**	51.3	40.26	42.5
Sylvester	19.6	13.2	9.3	10.2	15.8	16.3	8.3	8.3	15.9	16.9	**7.7**	7.9	12.8	14
Tiger2	17.7	17.2	18.1	16.2	16.5	16.1	17.4	19.1	*15.2*	16	19.3	19.1	15.7	**14.8**
Cow	32.1	30.6	**25.9**	26.9	27.1	28	29.2	28.6	27.4	27.1	28.7	29.6	28.7	*26.5*
Face	27.1	**20.3**	24.42	25.7	23.7	22.5	22.2	23	23.7	22.5	21.8	20.7	*20.5*	20.8
Twinings	11.9	*10.3*	11.6	11.2	12.8	13	10.7	10.8	11.4	12	**9.9**	11.1	11.3	11.4
sunshade	20.3	15.7	11.9	*11.7*	16.5	18.3	13.8	12.9	16.7	21	12.1	**11.4**	16.3	18.1
car	6.6	*4*	5.4	5.3	5.3	5.2	4.9	5	4.5	*4.1*	4.9	4.9	**3.8**	*4.1*
Average error	24.7	20.3	18.3	18.9	21	20.9	17.7	*17.5*	20.4	21.6	**15.7**	18.3	18.4	18.5
Average SD	12.2	7.5	6.9	8.7	8.9	9.8	7.2	6.8	8.5	9	2.5	4.9	6.8	7.7

Table 2: Center location error (CLE) and standard deviation for single and multiple classifier tracking

The results clearly demonstrate the improvement in performance when using classifier ensembles compared to a single classifier. It is also clear that as the size of the ensemble increases, the error is reduced. Table 2 also shows that the ensemble approach outperforms directly combining the features for a single classifier. This is true for all sequences except Face. Moreover, $Single_{500}$ is often outperformed even by ensembles of size 3 and 5 which use at most half of the features used by the single classifier. In very rare cases some ensemble approaches are outperformed by the single classifier. For example for Tiger 2 sequence, the ensembles with dynamic learning rates may degrade the performance while the ensembles with static learning rates outperform the single classifier. This is attributed to the addition of rapidly adaptive classifiers which are not suitable for the dataset and cause the tracker to drift towards the background. Therefore, as the ensemble grows, the error becomes higher because the ensemble is comprised of more classifiers with low learning rates. On the other hand the Twinings sequence gives better results with dynamic learning rates than static ones. Table 2 shows that using more diverse ensembles with varying learning rates generally yields better performance and has a lower average error than static ensembles. However there is no clear pattern demonstrating the effect of combining the classifiers with different values of C.

It is important to note that the standard deviation of $Single_{50}$ is very high compared to that of $Single_{500}$. This is due to the looser representation provided by the lower dimensional projections, which makes it suitable for MCS. It is also noteworthy that the ensemble approach significantly reduces the standard deviation of the error and that it becomes smaller as the size of the ensemble increases due to increased robustness to individual errors.

Figure 1 shows qualitative results for diverse ensemble tracking. Red bounding boxes represent the predictions of individual classifiers while green bounding boxes show the decision of the ensemble tracker. The results show that the tracker can adapt to rapid transformations such as scaling, rotation and off plane rotation which significantly changes the appearance of the object. By observing the decisions of the individual classifiers and the fused decision, it is clear that the ensemble approach is robust against outliers (which may result because the classifier is sensitive to the random matrix) and that the final decision is more accurate than the average of the individual decisions. Moreover, Figure 1 shows that a diverse ensemble with different learning rates can cope with different situations such as occlusion and 3D rotation. In the lemming sequence, the object is totally obscured in sample (b); sample (c) shows the target being reacquired on its reappearance while one classifier is adapted to the background due to the absence of the object. Sample (d) shows the highly adaptive classifiers keeping track of the object during off plane rotation and scaling while less adaptive classifiers drift towards the background. On the other hand, in the cow sequence in sample (b), the classifiers with low learning rate rapidly adapt to the surroundings and drift towards the hand, while the less adaptive classifiers keep track of the object accurately. This shows that different learning rates enable the tracker to cope with different situations and that the ensemble is tolerant to faults from different classifiers.

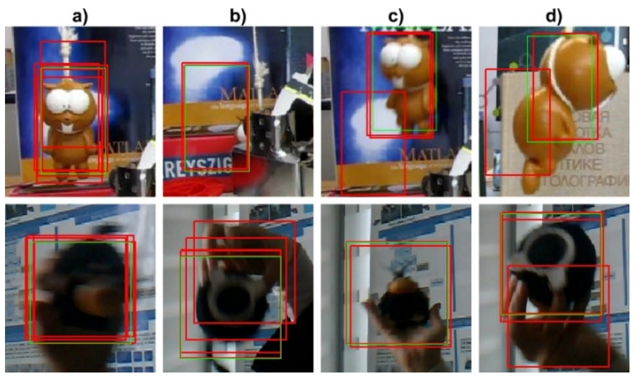

Figure 1: Samples for tracking results on 2 sequences.

6. Conclusions and Future Work

This paper proposes a robust multiple classifier approach to tracking by detection. The proposed approach provides the robustness associated with ensemble classifiers without needing additional information than that available for a single classifier. The conditions of sufficiency and diversity required for training classifier ensembles are satisfied by applying different sparse random matrices to project feature sets.

Individual classifier predictions are fused using a rank based method that employs the consensus of the ensemble members to alleviate independent errors. Quantitative experiments show significant improvement in error and variance over single classifiers. The ensemble approach is also shown to outperform directly combining multiple features sets. The tracking ensemble is real time and can be further merged with different tracking modules. In the future we aim to use the ensemble as a member of a multi method tracking ensemble. We also intend to adapt the parameters of the tracker automatically to suit the dataset.

References

[1] Breitenstein, M. D., Reichlin, F., Leibe, B., Koller-Meier, E., & Van Gool, L. (2009, September). Robust tracking-by-detection using a detector confidence particle filter. In Computer Vision, 2009 IEEE 12th International Conference on (pp. 1515-1522). IEEE.

[2] Andriluka, M., Roth, S., & Schiele, B. (2010, June). Monocular 3d pose estimation and tracking by detection. In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on (pp. 623-630). IEEE.

[3] Stalder, S., Grabner, H., & Van Gool, L. (2010). Cascaded confidence filtering for improved tracking-by-detection. In Computer Vision–ECCV 2010 (pp. 369-382). Springer Berlin Heidelberg.

[4] Kalal, Z., Mikolajczyk, K., & Matas, J. (2012). Tracking-learning-detection. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 34(7), 1409-1422.

[5] Grabner, H., Grabner, M., & Bischof, H. (2006, September). Real-Time Tracking via On-line Boosting. In BMVC (Vol. 1, No. 5, p. 6).

[6] Kim, T. K., Woodley, T., Stenger, B., & Cipolla, R. (2010, June). Online multiple classifier boosting for object tracking. In Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on (pp. 1-6). IEEE.

[7] Grabner, H., & Bischof, H. (2006, June). On-line boosting and vision. In Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on (Vol. 1, pp. 260-267). IEEE.

[8] Avidan, S. (2007). Ensemble tracking. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 29(2), 261-271.

[9] Visentini, I., Kittler, J., & Foresti, G. L. (2009). Diversity-based classifier selection for adaptive object tracking. In Multiple Classifier Systems (pp. 438-447). Springer Berlin Heidelberg.

[10] Minku, L. L., White, A. P., & Yao, X. (2010). The impact of diversity on online ensemble learning in the presence of concept drift. Knowledge and Data Engineering, IEEE Transactions on, 22(5), 730-742.

[11] Nishida, K. Y. O. S. U. K. E., & Yamauchi, K. O. I. C. H. I. R. O. (2007, August). Adaptive classifiers-ensemble system for tracking concept drift. In Machine Learning and Cybernetics, 2007 International Conference on (Vol. 6, pp. 3607-3612). IEEE.

[12] Triesch, Jochen, Christoph von der Malsburg, and Ruhr-Universitфt Bochum. "Self-Organized Integration of Adaptive Visual Cues for Face Tracking."

[13] Triesch, J., & Von Der Malsburg, C. (2001). Democratic integration: Self-organized integration of adaptive cues. Neural computation, 13(9), 2049-2074.

[14] Wu, Y., & Huang, T. S. (2004). Robust visual tracking by integrating multiple cues based on co-inference learning. International Journal of Computer Vision, 58(1), 55-71.

[15] Spengler, M., & Schiele, B. (2003). Towards robust multi-cue integration for visual tracking. Machine Vision and Applications, 14(1), 50-58.

[16] Isard, M., & Blake, A. (1998). Condensation—conditional density propagation for visual tracking. International journal of computer vision, 29(1), 5-28.

[17] Kwon, J., & Lee, K. M. (2010, June). Visual tracking decomposition. In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on (pp. 1269-1276). IEEE.

[18] Stalder, S., Grabner, H., & Van Gool, L. (2009, September). Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition. In Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on (pp. 1409-1416).

[19] Santner, J., Leistner, C., Saffari, A., Pock, T., & Bischof, H. (2010, June). Prost: Parallel robust online simple tracking. In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on (pp. 723-730). IEEE.

[20] Siebel, N. T., & Maybank, S. (2002, May). Fusion of multiple tracking algorithms for robust people tracking. In ECCV (4) (pp. 373-387).

[21] Kuncheva, L. I. (2003). That elusive diversity in classifier ensembles. In Pattern Recognition and Image Analysis (pp. 1126-1138). Springer Berlin Heidelberg.

[22] Hansen, L. K., & Salamon, P. (1990). Neural network ensembles. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 12(10), 993-1001.

[23] Li, Q., Wang, X., Wang, W., Jiang, Y., Zhou, Z. H., & Tu, Z. (2013, January). Disagreement-Based multi-system tracking. In Computer Vision-ACCV 2012 Workshops (pp. 320-334). Springer Berlin Heidelberg.

[24] Blum, A., & Mitchell, T. (1998, July). Combining labeled and unlabeled data with co-training. In Proceedings of the eleventh annual conference on Computational learning theory (pp. 92-100). ACM.

[25] Balcan, M. F., Blum, A., & Yang, K. (2004). Co-training and expansion: Towards bridging theory and practice. In Advances in neural information processing systems (pp. 89-96).

[26] Ho, T. K. (1998). The random subspace method for constructing decision forests. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 20(8), 832-844.

[27] Lai, C., Reinders, M. J., & Wessels, L. (2006). Random subspace method for multivariate feature selection. Pattern Recognition Letters, 27(10), 1067-1076.

[28] Skurichina, M., & Duin, R. P. (2001). Bagging and the random subspace method for redundant feature spaces. In Multiple Classifier Systems (pp. 1-10). Springer Berlin Heidelberg.

[29] Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

[30] Nigam, K., & Ghani, R. (2000, November). Analyzing the effectiveness and applicability of co-training. In Proceedings of the ninth international conference on Information and knowledge management (pp. 86-93). ACM.

[31] Slivka, J., Zhang, P., Kovacevic, A., Konjovic, Z., & Obradovic, Z. (2012, December). Semi-Supervised Learning on Single-View Datasets by Integration of Multiple Co-Trained Classifiers. In Machine Learning and Applications (ICMLA), 2012 11th International Conference on (Vol. 1, pp. 458-463). IEEE.

[32] Achlioptas, D.: Database-friendly random projections: Johnson-Lindenstrauss with binary coins. J. Comput. Syst. Sci 66, 671–687 (2003)

[33] Zhang, K., Zhang, L., & Yang, M. H. (2012). Real-time compressive tracking. In Computer Vision–ECCV 2012 (pp. 864-877). Springer Berlin Heidelberg.

[34] Baraniuk, R., Davenport, M., DeVore, R., & Wakin, M. (2008). A simple proof of the restricted isometry property for random matrices. Constructive Approximation, 28(3), 253-263.

[35] Babenko, B., Yang, M. H., & Belongie, S. (2011). Robust object tracking with online multiple instance learning. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 33(8), 1619-1632.

[36] Nebehay, G., Chibamu, W., Lewis, P. R., Chandra, A., Pflugfelder, R., & Yao, X. (2013). Can Diversity amongst Learners Improve Online Object Tracking?. In Multiple Classifier Systems (pp. 212-223). Springer Berlin Heidelberg.