Handling Uncertain Tags in Visual Recognition

Arash Vahdat and Greg Mori
School of Computing Science, Simon Fraser University, Canada
{avahdat, mori}@cs.sfu.ca

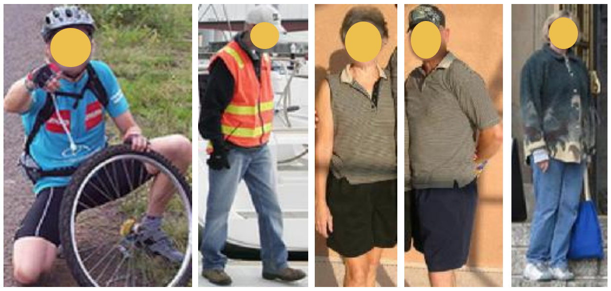
Abstract

Gathering accurate training data for recognizing a set of attributes or tags on images or videos is a challenge. Obtaining labels via manual effort or from weakly-supervised data typically results in noisy training labels. We develop the FlipSVM, a novel algorithm for handling these noisy, structured labels. The FlipSVM models label noise by "flipping" labels on training examples. We show empirically that the FlipSVM is effective on images-and-attributes and video tagging datasets.

1. Introduction

We present a novel algorithm for predicting a set of tags or attributes that describe an image or video. In recent years, there has been a push to broaden the scope of visual recognition – expanding the problem to consider describing images and videos rather than naming a single category. This push has been led by landmark work such as the "attributes" of Farhadi et al. [5] which framed the recognition problem as one of describing an image with a large set of attributes (e.g. shiny, red) in addition to a class label (apple).

A substantial body of related work falls into this vein, consisting of recognition of attributes for objects or keywords/tags for videos. Successfully describing images and videos will not only help to more accurately recognize image and video categories, but it will also provide a deeper understanding of the classified content from different perspectives such as quantity, quality, color, shape, parts, etc. This deeper understanding of images and videos can be used in a wide range of applications such as automatic description generation [10], face verification [11], image search [20] and video search [26].



	a)	b)	c)	d)	e)
Hand:	✓	✓	✗	✗	✗
Leg:	✓	✓	✗	✓	✓

Figure 1: Obtaining noise-free annotations is typically challenging. Attribute annotations for some images of people in the a-Pascal attribute dataset are shown. Note how the labeling of "Leg" in c) and d) or the labeling of "Hand" in b) and e) is inconsistent.

However, there are two important intrinsic problems in tag-based recognition: (1) tag annotations are noisy and (2) tags can be very difficult to observe from visual features. The algorithm we propose aims to address these issues.

There are two standard approaches to obtain tag annotations, either relying on human annotation or imputing them from weakly labeled data. Both are noisy processes.

Human annotation is subjective and many tags correspond to a very fine level of detail on a given image or video. Hence, human annotators may have different opinions about the presence of a tag. A typical solution to this problem is to ask several annotators to tag content followed by a tag aggregation approach. This process can reduce annotation noise but it is costly and remains error-prone. Consider the examples in Fig. 1 from the a-Pascal dataset [5].

Weakly labeled data are often used to obtain tag annotations. For instance, social media or multimedia sharing websites such as Flickr or YouTube can be used to extract a large volume of weakly annotated visual data. These data sources are often of poor quality, with many users entering a small subset of tags or spam tags for a given image/video.

This work was supported by NSERC and the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20069. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DOI/NBC, or the U.S. Government.

Beyond this, many tags are very difficult to discern visually. Again, considering the examples in Fig. 1, it is difficult to build classifiers from visual features to correctly predict all the labeled tags. Different tags have different degrees of difficulty and often there is a need to rely on contextual information or high-level reasoning.

In this paper we attack the problem of image and video classification based on uncertain tags. This is an essential issue to allow the recovery of detailed image or video descriptions from the typically ambiguous and noisy tag training data available. We present a novel structured tag prediction learning algorithm that considers the uncertainty of tags and their inter-relations.

We demonstrate the effectiveness of this algorithm via two sets of experiments. We show that modeling label noise improves performance in attribute-based image classification on the a-Pascal and a-Yahoo datasets [5] – datasets where ground truth is obtained from human annotations. Second, we automatically extract tag annotations for the TRECVID MED11 video dataset based on noisy, weakly supervised video description data. We apply our algorithm to learn tags from these data, and show they improve performance for classifying complex events.

2. Related Work

A substantial body of recent work considers the problem of labeling images with a set of tags or attributes. Farhadi et al. [5] train discriminative attribute classifiers that can be used to recognize different object classes, and further generalize to describe previously unseen categories of objects. Russakovsky and Fei-Fei [19] use transfer learning for similar recognition on large-scale object category datasets. Kumar et al. [11] use both comparative attributes (a mouth that looks like Barack Obama's) along with appearance attributes (gender, age, etc.) to improve face verification. Parikh and Grauman [15] propose an approach to discover a set of discriminative and nameable attributes with a human-in-the-loop framework. This line of work was expanded to relative attributes [16], recognizing the strength of an attribute by learning a ranking function.

Related work also exists in the video analysis domain, tagging videos with relevant keywords or concepts. Yang and Toderici [26] propose a latent model to train classifiers for sub-categories of tags. Qi et al. [17] use structural SVM to predict multiple tags while considering their correlation.

Weakly labeled data are often used to extract tag or attribute data for images or videos. Ferrari and Zisserman [7] propose a generative probabilistic model to recognize low level attributes such as "red" or "striped" from weakly labeled images obtained from the internet. Berg et al. [1] propose an approach to extract common attributes for certain objects by mining the text associated with images. Similarly, Leung et al. [12] have proposed to use Multiple Instance Boosting [23] to train video tag classifiers using noisy web videos. All these methods consider the presence of noise in the training data. However, they learn a classifier for each attribute independently and ignore the fact that tags are highly correlated. Our algorithm develops a principled max-margin based criterion for learning with structured, noisy tags.

A wide range of machine learning tools has been used for training tag classifiers. Structural SVM [21] is used in Siddiquie et al [20] for multi-attribute image ranking and retrieval. Latent SVM [6] is used for object and video classification respectively by Wang and Mori [25] and Izadinia and Shah [8] in which tags are modeled as latent variables.

In the line of robust classifiers, RampSVM [3] has been proposed to train SVM classifiers that are robust against some amount of annotation noise. Ramp loss has been extended recently to the structural prediction case by McAllester and Keshet [13] and Chapelle et al. [2]. None of these works have been explored on the problem of image or video tagging. The objective function of Structural RampSVM is similar to ours in the sense that the training annotation is not assumed to be completely accurate. However, in Structural RampSVM the risk is still measured with respect to the training annotation, while in our method we compute the risk with respect to refined labels.

3. Flip Support Vector Machine

Our goal in this paper is to recognize a set of predefined classes in images or videos. For this purpose, we are interested in considering tags which represent different aspects of classes. Ultimately, recognition of these tags will provide us with more description of visual content. In order to train a classifier, we are going to use a training data set of images and videos that are annotated with their class and tag labels. In this section, our approach for training the classifier from noisy tag labels is presented. Tag labels are often noisy, due to the large-scale manual annotation effort required or use of ad-hoc techniques on weakly labeled data to extract tags. Here, tags may correspond to attributes of objects or tags/key-words of videos.

For tag-based classification, we will use a structured model similar to Wang and Mori [25] that considers the interaction between tags and class labels. Interaction between tags helps us to recognize tags that are correlated. For example, "has eye", which is highly correlated with "has head", is very difficult to detect in isolation. But, eye detection can be improved by considering the presence of head. Similarly, classification can be enhanced considering the interaction between tags and class labels. For example, an object that has "clothes" and "skin" is more likely a human rather than a car.

In our model, a sample is represented with a triple $(x, \mathbf{t}, y)$. x represents an image or a video, $\mathbf{t} = \{t_i\}_{i=1}^{i=T}$ represents the presence/absence of T tags with binary labels, $t_i \in \{-1, 1\}$. If the ith tag is present in the image x, we will have $t_i = 1$, otherwise $t_i = -1$. Similarly, $y \in \{-1, 1\}$ is a binary variable that shows the presence/absence of a class.

The goal of training is to learn a scoring function that can predict the class of an example in test time by considering both visual features and the presence/absence of tags. Here, the scoring function is defined as $F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{R}$ that scores an example and a class label, $y \in \mathcal{Y}$ where $\mathcal{Y} = \{-1, 1\}$ for our binary case. In test time, an example will be assigned to the class that has highest score, $y^* = \arg\max_{y \in \mathcal{Y}} F(x, y)$. Similar to Latent SVM framework [6, 24] or Structural SVM [21], it is assumed that $F(x, y) = \max_{\mathbf{t}} w^T \Phi(x, \mathbf{t}, y)$ where $w^T \Phi(x, \mathbf{t}, y)$ is a linear potential function that scores a configuration of tag labels and a class label for a sample considering their interaction. The tag dependency is modeled using a tree-structured graph called *tag interactions graph*.

Next, our proposed learning framework for training parameters of a structured model from noisy tags is presented. We defer comparisons to other learning criteria and model details to Sec. 3.2 and Sec. 3.3 respectively.

3.1. Training a Flip SVM

The main contribution of our paper is to develop a novel max-margin approach that enables us to train the model parameters from noisy labeled data. The idea of our approach is that the learning algorithm is allowed to change training tag labels while penalizing the number of changes. This way the algorithm may be able to correct some tag label mistakes but it is limited not to change all the labels arbitrarily. We call this training framework *FlipSVM (FSVM)* as it flips some of the labels in the course of training.

Given a set of N examples, $\{(x_n, \mathbf{t}_n, y_n)\}_{n=1}^{n=N}$ for training, we propose the optimization of FlipSVM as:

$$\min_{w, \xi_n, \xi'_n, \mathbf{t}'_n} \frac{\lambda}{2}||w||_2^2 + \sum_{n=1}^{N} \xi_n + \gamma \sum_{n=1}^{N} \xi'_n$$

$$\text{s.t.} \quad \xi'_n \geq \Delta'_{\mathbf{t}'_n, \mathbf{t}_n} \tag{1}$$

$$w^T \phi(x_n, \mathbf{t}'_n, y_n) \geq w^T \phi(x_n, \mathbf{t}, y) + \Delta^{y, y_n}_{\mathbf{t}, \mathbf{t}'_n} - \xi_n \quad \forall \mathbf{t}, \forall y$$

that minimizes the norm of parameters ($||w||_2^2$), structured error ξ_n and the tag label change cost ξ'_n computed over training examples. Here the refined tag labels $\mathbf{t}'_n = \{t'_{ni}\}_{i=1}^{i=T}$ are introduced for the nth training example in order to measure the error respect to them instead of the noisy ground truth labels. Therefore, the goal of training is to find a w that produces a score for refined tag labels $\mathbf{t}'_n$ and ground truth class label y_n greater than any other hypothesized labeling with a margin re-scaled with the loss function $\Delta^{y, y_n}_{\mathbf{t}, \mathbf{t}'_n}$ that measures the *badness* of the hypothesized labeling.

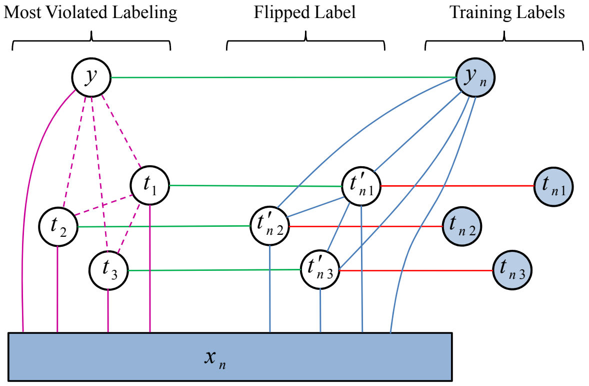


Figure 2: Min-max optimization in FlipSVM for decomposable $\Delta^{y, y_n}_{\mathbf{t}, \mathbf{t}'_n}$ and $\Delta'_{\mathbf{t}'_n, \mathbf{t}_n}$ and a tree-structured tag interaction graph. Shaded variables represent observed variables and different colored links denote potential function defined in different terms of Eq. 3: $w^T \phi(x_n, \mathbf{t}, y)$, $\Delta^{y, y_n}_{\mathbf{t}, \mathbf{t}'_n}$, $w^T \phi(x_n, \mathbf{t}'_n, y_n)$, and $\Delta'_{\mathbf{t}'_n, \mathbf{t}_n}$. By simplifying the model and ignoring pairwise terms (dashed lines) in most violated labeling inference, maximization over $\mathbf{t}$ decomposes to maximization over each t_i that depends on individual t'_{ni}.

$\Delta'_{\mathbf{t}'_n, \mathbf{t}_n}$ measures the cost of label flips, and similar to $\Delta^{y, y_n}_{\mathbf{t}, \mathbf{t}'_n}$ is assumed to be a function that can be decomposed to a sum of losses measured on individual output variables. γ and λ are the trade-off parameters that tune the impact of model complexity and label flip cost. We can write an unconstrained version of the optimization in Eq. 1 as:

$$\min \frac{\lambda}{2}||w||_2^2 + \sum_{n=1}^{N} R_n(w) \tag{2}$$

where $R_n(w)$ is the the risk function:

$$R_n(w) = \min_{\mathbf{t}'_n} \max_{y, \mathbf{t}} \Big(w^T \phi(x_n, \mathbf{t}, y) + \Delta^{y, y_n}_{\mathbf{t}, \mathbf{t}'_n} \tag{3}$$
$$- \quad w^T \phi(x_n, \mathbf{t}'_n, y_n) + \gamma \Delta'_{\mathbf{t}'_n, \mathbf{t}_n} \Big)$$

Training of a FlipSVM consists of two parts. First, the risk function should be computed for each example given the model parameters, w. Second, the model parameters should be updated given the risk values. The following explains these two steps in detail.

Risk Evaluation: The optimization problem in Eq. 3 is min-max optimization of a Markov network. The inner maximization finds the most violated labeling and the outer minimization refines the ground truth label such that sum of the most violated labeling margin and the flip cost is minimum. Even in our simple case shown in Fig. 2, where $w^T \phi(x, \mathbf{t}, y)$ is defined on a tree structured tag interaction graph, this optimization problem is an NP-hard problem.

For solving this optimization problem, we propose a simple efficient heuristic. The idea of our heuristic is to form an approximate version of the min-max optimization that can be solved exactly. If the scoring function used for finding the most violated labeling in Eq. 3 is simplified by omitting its pairwise terms, the inner maximization can be solved independently for each t_i given t'_{ni}. The optimum value of the simplified maximization can be considered as a potential term that depends on t'_{ni} (unary term). Given these values, our minimization becomes so-called loss augmented [21] inference that can be solved exactly by dynamic programming for a tree structured graph.

The relaxation approach to the inner maximization provides us with an approximated refined labeling, t^*_n. But, the value of the risk function at t^*_n should also be computed. For this purpose, the risk is evaluated by plugging t^*_n in Eq. 3 and optimizing the inner maximization on the full model with all terms using the same dynamic programming that was used to produce t^*_n.

In our risk computation, loss augmented inference is solved twice, first for finding approximate $\mathbf{t}^*_n$ and then for solving the inner maximization. As the number of tags is typically on the order of tens, and due to the structure of the tag interaction graph, the inference problem can be solved very efficiently.

Optimizing w: The optimization over w in Eq. 2 is a non-convex optimization problem. Here, we use the NRBM method proposed by Do and Artières [4] to optimize w. NRBM is a non-convex extension of the Cutting Plane algorithm, and is used for training latent SVM [25]. This technique requires us to compute the value of the risk and its gradient with respect to w. In a nutshell, this algorithm creates an approximation of the original quadratic programming in Eq. 2 by iteratively adding a cutting plane at the current optimum and updating the optimum.

3.2. FSVM vs. Structural SVM and Latent SVM

Structural SVM (SSVM) [21] and Latent SVM (LSVM) [27] are two commonly used max-margin approaches that could be deployed for training a tag-based classifier. SSVM trains the parameters of model such that both the class and tag labels of the training data can be predicted accurately. In the presence of noisy tags, SSVM does not model the label noise and it may fit to noisy tag labels.

In contrast, noisy tag labels can be modeled as latent variables using a Latent SVM model. This method does not have any notion of loss on latent variables, and it ignores all the information about the ground truth tags. In [25, 8], tag information is injected into training by replacing the image features with the score of pre-trained tag classifiers. Our approach is different from the Latent SVM approach, as it enables us to train the tag and label classifiers in an unified approach.

3.3. Model

In order to be able to compare our training algorithm with previous works on tag-based classification, we use a model similar to Wang and Mori [25] for scoring the configuration of tag labels and class label. Their work has been later adopted by Izadinia and Shah [8] to recognize a set of complex events in videos by considering noisy low-level events. In this section, we briefly review our model focusing on the differences with Wang and Mori's [25].

In our model, the dependency of the tags is represented using an undirected tag interaction graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the vertex set $\mathcal{V} = \{1, 2, 3, ..., T\}$ for all tags, and the edge set E in which $(i, j) \in \mathcal{E}$ indicates the inter-dependency of the ith and jth tag. Our scoring function measures the compatibility of a class label y and tag labels $\mathbf{t}$ in an example x by:

$$w^T\phi(x, \mathbf{t}, y) = y\theta^T\phi(x) + \sum_{i\in\mathcal{V}} t_i\alpha_i^T\phi(x) \qquad (4)$$
$$+ \sum_{(i,j)\in\mathcal{E}} \beta_{ij}^T\varphi(t_i, t_j) + \sum_{i\in\mathcal{V}} \nu_i^T\varphi(t_i, y)$$

where $w = \{\theta, \alpha_i, \beta_{ij}, \nu_i\}_{i\in\mathcal{V},(i,j)\in\mathcal{E}}$.

The scoring function has four parts: The potential terms $y\theta^T\phi(x)$ and $t_i\alpha_i^T\phi(x)$ measure the compatibility of a global feature $\phi(x)$ extracted from example x with class or tag labels. As we are considering a binary classifier we learn θ and α_i parameters that separate positive class from the negative class by assigning them to positive values. $\beta_{ij}^T\varphi(t_i, t_j)$ measures the compatibility between a pair of tags that is connected in our tag interaction graph $\mathcal{E}$. $\varphi(t_i, t_j)$ is a sparse vector of size four that has only a 1 value indicating which of the cases in $\{(0, 1), (1, 0), (0, 0), (1, 1)\}$ is taken by (t_i, t_j). Similarly, $\nu_i^T\varphi(t_i, y)$ measures the compatibility between a tag and a class label.

In reality, only a small subset of tag pairs may show a high amount of correlation. Therefore, considering all pairs of tags will not be necessary, and would slow down inference significantly. Here, the same technique of [25, 17] is used to extract a sparse set of dependencies. Normalized mutual information is computed between each pair of tags, and a maximum spanning tree of edges is selected. By employing this approach, inference of tags is done quickly using dynamic programming. Note that, even in our case that tags are noisy, this approach considers statistics that are robust against noise when we are extracting tag dependencies.

Both [25] and [8] replace the long feature vector $\phi(x)$ with the output of a binary tag/class SVM classifier. In our case training tags are noisy, so pre-trained binary classifiers will not be immune to noise. Therefore, we do not pretrain a classifier, and we let our training algorithm train these models all together. Wang and Mori [25] have also an extra class specific tag model. For our binary classification, this part has been excluded.

4. Experiments

We evaluated our approach on two different tasks. First, we consider the problem of attribute-based object classification in the a-Pascal and a-Yahoo datasets [5]. Second, we examine the problem of tag-based video classification from weakly supervised (and hence noisy) tags. We automatically extract a set of video tags by processing the text description files provided in TRECVID MED11 [14] videos.

Evaluation Methodology: In all experiments, we will train a binary classifier for recognizing classes (object classes or video classes). We evaluate using Average Precision (AP), a standard performance measure based on area under a precision-recall curve. Generating this curve requires a score representing the confidence of our model in assigning a sample to a class. Computing this score for SSVM and FSVM is not trivial, as these models return a configuration of labels that has highest score for a test example. In order to extract the confidence score for labeling an example, we apply a softmax function to best scoring label configurations for each class. First, the label score for the example x is computed by:

$$s(y) = \max_{\mathbf{t}} w^T \phi(x, \mathbf{t}, y) \quad \forall y \in \mathcal{Y} \tag{5}$$

Then, the probability of each class is defined as $P(y) = \frac{exp(s(y))}{z_p}$ where $z_p = \sum_{y \in \mathcal{Y}} exp(s(y))$. A similar approach is used by [17] to compute tag indexing scores from SSVM.

4.1. Object Classification

In this section, we utilize our model for classifying objects from attributes in the a-Pascal and a-Yahoo image datasets [5]. The a-Pascal dataset has 6340 training images and 6355 test images (from PASCAL VOC 2008) annotated with 64 binary attributes such as “2D Boxy”, “Shiny”, “Has leg” etc. These attributes belong to three categories including: shape, material, and part. Each image is assigned to one of 20 object classes including: aeroplane, bicycle, bird, boat, bottle, bus, car, cat, chair, cow, dining table, dog, horse, motorbike, person, potted plant, sheep, sofa, train and tv/monitor. The a-Yahoo dataset has 2644 images annotated with the same set of attributes. However, the object classes are different, including: bag, building, carriage, centaur, donkey, goat, jetski, monkey, mug, statue, wolf and zebra. We use the global features provided by Farhadi et al. [5]. Each image is represented with a 9751-dimensional feature vector that contains information on color, texture, visual words, and edges. In this section, we utilize object attributes as tags for the object classification task.

Modeling Noisy Labels: In this experiment we measure FlipSVM under the original (often noisy) tag labels, and examine the performance of our proposed approach under different amounts of additional noise. We run experiments using the original a-Pascal and a-Yahoo tag annotations, and further introduce different levels of synthetic noise to the ground truth tag labels. In order to generate noise, we follow the approach of Leung et al. [12] which generates noise for positive tags by changing the label of some samples with negative tag to positive. Leung et al. argue this is a good model of label noise, basically since negative data are common and the noise level in them negligible.

We examine our approach under three different noise levels: 0%, 16% and 33%. In $X\%$ noise, $X\%$ of the training samples with each positive tag are mislabeled samples added from the negative set.

Dataset Split: We use the standard train/test split for the a-Pascal dataset. We further sub-divide the training set to create a validation set (25% of the training images) for tuning the parameters of our model and all other baselines. As a binary classification problem is considered here, parameters are tuned for each class separately on the validation set. A train/test split is not provided for the a-Yahoo dataset. 37.5%, 12.5% and 50% of images are selected for training, validation and testing respectively for this dataset.

Loss Functions: We use re-scaled hamming loss for both $\Delta^{y,y_n}_{\mathbf{t},\mathbf{t'_n}}$ and $\Delta'_{\mathbf{t}'_n,\mathbf{t}_n}$. For each tag or class, re-scaling of the loss function is performed based on the number of training samples that have that tag or class. $\Delta^{y,y_n}_{\mathbf{t},\mathbf{t'_n}}$ is defined as:

$$\Delta^{y,y_n}_{\mathbf{t},\mathbf{t'_n}} = \delta(y, y_n) + \sum_{i \in \mathcal{V}} \delta_i(t_i, t'_{ni}) \tag{6}$$

where

$$\delta(y, y_n) = \begin{cases} \frac{C}{N_y}, & y \neq y_n \\ 0, & y = y_n \end{cases} \qquad \delta_i(t_i, t'_{ni}) = \begin{cases} \frac{1}{N^{(i)}_{t'_{ni}}}, & t_i \neq t'_{ni} \\ 0, & t_i = t'_{ni} \end{cases}$$

N_y is the number of training examples that belongs to class $y \in \mathcal{Y}$. Similarly, $N^{(i)}_{t'_{ni}}$ is the number of training images whose ith tag is $t'_{ni} \in \{-1, 1\}$. Since we are interested in classifying images to object classes rather than tag classes, we set $C = 25$ to have a loss function more sensitive to class than tag error.

$\Delta'_{\mathbf{t}',\mathbf{t}}$ is also defined as re-scaled hamming loss which basically counts the number of label changes. We here rescale the hamming loss such that tags with a large number of positive examples are penalized less. This way, we encourage our training algorithm to change frequent tags more. We also prevent label flips from a negative tag to a positive tag. The loss function becomes:

$$\Delta'_{\mathbf{t}'_n,\mathbf{t}_n} = \sum_{i \in \mathcal{V}} \delta'_i(t'_{ni}, t_{ni}) \colon \quad \delta'_i(t'_{ni}, t_{ni}) = \begin{cases} \infty & t'_{ni} = 1, t_{ni} = -1 \\ \frac{1}{N^{(i)}_{t_{ni}}} & t'_{ni} = -1, t_{ni} = 1 \\ 0 & t'_{ni} = t_{ni} \end{cases}$$

Note that the loss functions as well as the tag interaction graph are formed from noisy training tags for each experiment and noise level separately.

Baseline Methods: For each experiment, we compare our method with three strong baselines: (1) Structural SVM that uses the same model and loss function, Δ as ours. (2) Latent SVM that uses our model equipped with a class-specific tag model discussed in [25]. (3) SVM classifier trained on global features ignoring all the information about tag annotation. The Latent SVM baseline is the method in [25], re-implemented for the binary classification version to be examined under different noise levels.

Classification Results: Table 1 report mean AP (mAP) of our method compared with three baselines on the a-Pascal and a-Yahoo datasets. Mean AP is computed by taking the average of AP over all object classes in these datasets. Several observations can be made. First, it can be easily seen that our proposed FlipSVM achieves better or comparable results when there is no additional noise added to the annotations – likely because it handles ambiguity and noise inherent in the dataset labels. As we increase label noise, the performance of both Latent SVM and Structural SVM start to fall while FlipSVM shows robustness against training data noise.

Label Flip Results: We further measure the quality of the label flips that our algorithm produces on the training data, as we have access to the ground truth tag labels before and after adding synthetic noise. We measured precision and recall of our algorithm's label flips at the end of learning. A true positive label flip is defined as a change in label that changes a noisy tag label back to the original ground truth label. On average, our algorithm's flips have 73.30% precision and 19.20% recall for the a-Pascal dataset and 69.72% precision and 19.99% recall for the a-Yahoo dataset in the experiments at 33% noise level. These numbers show that our algorithm flips labels conservatively but with high precision. Note that chance performance is 33% precision for any recall.

4.2. Video Classification

In this experiment, we use our model to classify complex events in web videos using the TRECVID MED11 dataset [14]. We follow the standard evaluation protocol. The dataset contains 15 events that are divided across two collections, DEV-T and DEV-O. The DEV-T dataset consists of 10,723 videos including videos from five event categories: *board trick (E1)*, *feeding animal (E2)*, *landing fish (E3)*, *wedding ceremony (E4)*, and *woodworking project (E5)*. The DEV-O collection is significantly larger, 32,061 videos, and includes ten categories: *birthday party (E6)*, *changing a tire (E7)*, *flash mob (E8)*, *getting a vehicle unstuck (E9)*, *grooming animal (E10)*, *making sandwich (E11)*, *parade (E12)*, *parkour (E13)*, *repairing appliance (E14)*, and *sewing project (E15)*. Both DEV-T and DEV-O are dominated by videos of the null category (i.e., background videos that do not contain the events of interest). For training, an Event-Kit data collection, containing roughly 150 positive videos per category, is also provided.

Table 1: Object classification results on a-Pascal and a-Yahoo datasets. Number denote mean AP in %. Our method is compared with three strong bases lines. Our training algorithm shows robustness against noise in tag annotations.

a-Pascal

Noise	**0%**	**16%**	**33%**
SVM	37.05	37.05	37.05
Latent SVM [25]	37.65	37.22	36.19
Structural SVM	38.67	38.46	37.76
FlipSVM	**40.17**	**40.41**	**39.53**

a-Yahoo

Noise	**0%**	**16%**	**33%**
SVM	62.52	62.52	62.52
Latent SVM [25]	64.58	63.73	63.46
Structural SVM	**67.14**	66.78	65.27
FlipSVM	66.37	**66.92**	**66.52**

Dataset Split and Feature: This experiment uses HOG3D features, k-means quantized into a 1,000 word codebook. For improving performance of all techniques, the Histogram Intersection Kernel is approximated using feature extension [22]. Similar to the experiments in Sec. 4.1, a train/validation/test protocol is employed to tune the parameters of our model and all the baselines. A binary classifier is trained using a randomly sampled 80% of the Event Kit videos. For E1 to E5, test is done on DEV-T dataset, while the parameters are validated on the remaining 20% of Event Kit augmented with 10,000 randomly selected background videos from the DEV-O dataset. The same training data are used for experiments on E6 to E15. However, DEV-T and the remaining 20% of Event Kit videos are used for validation and DEV-O is used for test.

Collecting Video Tags: Obtaining tag-level annotation for this huge number of videos is very costly. Instead of manual labeling, we employ a simple technique to extract tags on this dataset. The advantage of our approach is that it is very fast, and it can be used to extract noisy tags on the whole dataset with no manual interaction. Moreover, the nature of the tags is similar to those tags that can be extracted by processing user-provided descriptions from social websites in the sense that a subset of present tags are extracted by our method (e.g. Kennedy et al. [9] show that 50% of annotated tags are actually in the Flickr images).

The TRECVID MED11 dataset includes "judgment files" that contain a short one-sentence description for each video. An example description is: "A man and a little boy lie on the ground after the boy has fallen off his bike.”. This sentence provides us with information about presence of objects such as “man”, “boy” and “bike” or actions such as “lying” or “falling off” in the sequence. This is very limited information as it does not provide an exhaustive tag set denoting the absence of all other tags, but it can still be used to train a tag classifier for the present objects or actions.

We use a simple approach to extract tags focusing on the objects in the videos. We use Topia [18], an open source text analysis tool, to extract frequent nouns in the judgment files. This software automatically performs part-of-speech tagging for each sentence to detect nouns, adjectives, verbs, etc. Then, a simple stemmer maps plural nouns to singular ones. Finally, noun frequencies are collected and those that are above a threshold are selected. This process results in 75 tags on the training dataset. Randomly selected examples of the tags are: dance, soccer, lake, river, road, snowboard, girl, street, kitchen, boat, rally, egg, car, etc. We also added 5 genres and 20 topic tags provided in judgment files to create a 100-tag set for the TRECVID MED11 dataset[1].

Comparison: For this dataset, we trained a tag-based event classifier using the same model that we presented in Sec. 3.3. This model has been used by Izadinia and Shah [8] for the same task. Unfortunately, direct comparison to this work is not possible as we do not have access to their features and tag annotation. Moreover, we test on the full dataset using the standard evaluation protocol, versus [8] which experiments on Event Kit videos.

We compared our method to the same three baselines that were used in Sec. 4.1. They are all trained with the same HOG3D features. The loss functions are defined slightly different to those in Sec. 4.1. In the TRECVID MED dataset, annotation is done by expert annotators. We assume that the annotators have not entered spam sentences and sentences are actually representing the content of video. So, the extracted tags are actually present in the video. However, sentences may not have all the tags that are in a video. Therefore, the label flip loss function is modified to prevent label flips from a positive tag to a negative tag.

Table 2 report the performance on the DEV-T and DEV-O datasets. Our model significantly outperforms the baseline methods. We conducted paired t-tests on the AP values across the two datasets to compare our FlipSVM against each baseline for the null hypothesis that their mAP is better than ours. The resulting p-values are 1.02%, 0.78% and 4.81%, all under 5% significance level for SVM, SSVM and LSVM respectively. Qualitative visualization of our results for three categories is also shown in Fig. 3.

Table 2: Performance comparison against baselines on DEV-T for E1-E5 and DEV-O for E6-E15. Numbers denote the average precision, in %. Best result for a particular event category is shown in bold.

DEV-T dataset

Events	SVM	SSVM	LSVM	FSVM
E1	13.30	14.15	14.76	**15.84**
E2	**3.90**	3.49	3.49	3.28
E3	15.69	13.30	15.69	**18.94**
E4	29.79	36.49	37.26	**38.27**
E5	9.53	8.36	10.92	**11.19**
mAP	14.44	15.16	16.42	**17.50**

DEV-O dataset

Events	SVM	SSVM	LSVM	FSVM
E6	**5.48**	4.41	5.41	4.57
E7	**3.85**	0.86	3.87	2.33
E8	26.13	27.76	25.22	**30.16**
E9	3.58	**3.64**	4.72	3.54
E10	1.17	1.31	1.24	**1.63**
E11	2.62	**3.29**	2.52	2.82
E12	5.95	6.53	8.09	**8.20**
E13	9.37	**13.77**	10.95	13.26
E14	8.14	14.72	8.65	**15.89**
E15	**2.04**	1.67	1.87	1.40
mAP	6.83	7.80	7.25	**8.38**

5. Conclusion

Label noise is an inherent problem in learning for visual recognition. The problem is especially acute for large-scale datasets with multiple output tags or attributes – which is becoming a common paradigm. With more complex labels come challenges in accurate annotation and varying degrees of difficulty in recognition. In this paper we presented the FlipSVM, a learning framework designed to address these challenges. We showed that the FlipSVM model for label noise can improve performance at image recognition in the presence of noisy attribute data and video classification from weakly supervised tag sets. Generating tags, attributes, or descriptions of images and videos is a promising research direction, and novel learning frameworks will be needed to make further progress.

References

[1] T. L. Berg, A. C. Berg, and J. Shih. Automatic attribute discovery and characterization from noisy web data. In *ECCV*, 2010. 2

[2] O. Chapelle, C. B. Do, Q. V. Le, A. J. Smola, and C. H. Teo. Tighter bounds for structured estimation. In *NIPS*, 2008. 2

[3] R. Collobert, F. H. Sinz, J. Weston, and L. Bottou. Trading convexity for scalability. In *ICML*, 2006. 2

[4] T. M. T. Do and T. Artières. Large margin training for hidden markov models with partially observed states. In *ICML*, 2009. 4

[5] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009. 1, 2, 5

[1] Tag annotations are available in authors’ website.

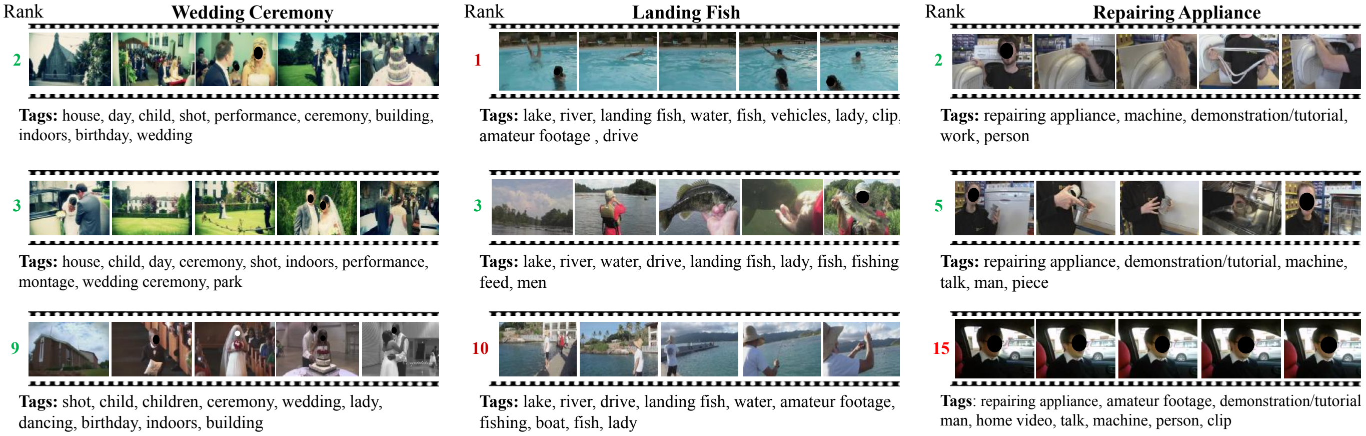

Rank
Wedding Ceremony
2
Tags: house, day, child, shot, performance, ceremony, building, indoors, birthday, wedding
3
Tags: house, child, day, ceremony, shot, indoors, performance, montage, wedding ceremony, park
9
Tags: shot, child, children, ceremony, wedding, lady, dancing, birthday, indoors, building
Rank
Landing Fish
1
Tags: lake, river, landing fish, water, fish, vehicles, lady, clip, amateur footage , drive
3
Tags: lake, river, water, drive, landing fish, lady, fish, fishing feed, men
10
Tags: lake, river, drive, landing fish, water, amateur footage, fishing, boat, fish, lady
Rank
Repairing Appliance
2
Tags: repairing appliance, machine, demonstration/tutorial, work, person
5
Tags: repairing appliance, demonstration/tutorial, machine, talk, man, piece
15
Tags: repairing appliance, amateur footage, demonstration/tutorial man, home video, talk, machine, person, clip

Figure 3: Qualitative visualization of TRECVID MED results. Each column shows three videos that are retrieved for a particular event category from the test dataset. The rank of each video is reported next to it, where its color indicates the ground truth event label, i.e. red/green indicate the negative/positive classes respectively. Tags are shown below each video sorted by decreasing confidence score. Here at most 10 tags that have score greater than a threshold are reported. Our algorithm not only learns the tags that are associated with each category, but also it discriminates some less common ones such as "park" and "dancing" in wedding ceremony, "feed" in landing fish, or "talk" in repairing appliance. Note how the detection of "river", "lake" and "water" misleads our algorithm to classify two background videos as landing fish category.

[6] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 32(9):1627–1645, 2010. 2, 3

[7] V. Ferrari and A. Zisserman. Learning visual attributes. In *NIPS*, 2007. 2

[8] H. Izadinia and M. Shah. Recognizing complex events using large margin joint low-level event model. In *ECCV*, 2012. 2, 4, 7

[9] L. S. Kennedy, S.-F. Chang, and I. Kozintsev. To search or to label?: predicting the performance of search-based automatic image classifiers. In *ACM Multimedia Information Retrieval*, 2006. 6

[10] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg. Baby talk: Understanding and generating simple image descriptions. In *CVPR*, 2011. 1

[11] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *ICCV*, 2009. 1, 2

[12] T. Leung, Y. Song, and J. Zhang. Handling label noise in video classification via multiple instance learning. In *ICCV*, 2011. 2, 5

[13] D. A. McAllester and J. Keshet. Generalization bounds and consistency for latent structural probit and ramp loss. In *NIPS*, 2011. 2

[14] P. Over, G. Awad, M. Michel, J. Fiscus, W. Kraaij, A. F. Smeaton, and G. Quenot. Trecvid 2011 — an overview of the goals, tasks, data, evaluation mechansims and metrics. In *Proceedings of TRECVID 2011*, 2011. 5, 6

[15] D. Parikh and K. Grauman. Interactively building a discriminative vocabulary of nameable attributes. In *CVPR*, 2011. 2

[16] D. Parikh and K. Grauman. Relative attributes. In *ICCV*, 2011. 2

[17] G.-J. Qi, X.-S. Hua, Y. Rui, J. Tang, T. Mei, and H.-J. Zhang. Correlative multi-label video annotation. In *ACM Multimedia*, 2007. 2, 4, 5

[18] S. Richter, R. Ferriday, and the Zope Community. Topia, content term extraction using POS tagging, https://pypi.python.org/pypi/topia.termextract, Jan. 2013. 7

[19] O. Russakovsky and F.-F. Li. Attribute learning in large-scale datasets. In *ECCV Worksh. on Parts and Attributes*, 2010. 2

[20] B. Siddiquie, R. Feris, and L. Davis. Image ranking and retrieval based on multi-attribute queries. In *CVPR*, 2011. 1, 2

[21] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *ICML*, 2004. 2, 3, 4

[22] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009. 6

[23] P. A. Viola, J. C. Platt, and C. Zhang. Multiple instance boosting for object detection. In *NIPS*, 2005. 2

[24] Y. Wang and G. Mori. Max-margin hidden conditional random fields for human action recognition. In *CVPR*, 2009. 3

[25] Y. Wang and G. Mori. A discriminative latent model of object classes and attributes. In *ECCV*, 2010. 2, 4, 6

[26] W. Yang and G. Toderici. Discriminative tag learning on youtube videos with latent sub-tags. In *CVPR*, 2011. 1, 2

[27] C.-N. J. Yu and T. Joachims. Learning structural SVMs with latent variables. In *ICML*, 2009. 4