A topological framework for training Latent Variable Models

Heydar Maboudi Afkham and Carl Henrik Ek and Stefan Carlsson

{heydarma,chek,stefanc}@csc.kth.se
Computer Vision and Active Preception Lab., KTH
SE-100 44, Stockholm, Sweden

***Abstract*—We discuss the properties of a class of latent variable models that assumes each labeled sample is associated with a set of different features, with no prior knowledge of which feature is the most relevant feature to be used. Deformable-Part Models (DPM) can be seen as good examples of such models. These models are usually considered to be expensive to train and very sensitive to the initialization. In this paper, we focus on the learning of such models by introducing a topological framework and show how it is possible to both reduce the learning complexity and produce more robust decision boundaries. We will also argue how our framework can be used for producing robust decision boundaries without exploiting the dataset bias or relying on accurate annotations. To experimentally evaluate our method and compare with previously published frameworks, we focus on the problem of image classification with object localization. In this problem, the correct location of the objects is unknown, during both training and testing stages, and is considered as a latent variable.**

I. Introduction

Latent variable models are well-known for their strength in automatically adapting to variations of the data. In this paper, we focus on a specific class of latent variable models for discriminative learning. These models assume that a set of feature vectors is associated with each labeled sample and the role of the latent variable is to select one feature vector from this set, to be used in the calculations. In both training and testing stages, these models assume that no prior knowledge is provided about which features are to be used. Deformable Part Models (DPM) [5], [6] can be seen as a good example of these models. With the aid of Latent SVM framework, DPM provides a level of freedom for samples, in terms of relocatable structures, to adapt to the intra-class variation. As the result of this flexibility, the appearance of the samples becomes more unified and the training framework can learn a more robust classifier over the training samples. A good practice of the model discussed in [5] can be found within the DPM framework. In their work, the method does not assume the ground truth bounding boxes are perfectly aligned and leaves it to the model to relocate the bounding boxes and find a better alignment between the samples. The location of this alignment is regarded as a latent variable. In a more complex example [12], [8], the task is to train an object detector without having prior knowledge of the location of the objects and considering it as a latent variable. In their work, it is left to the learning framework to both locate the objects in the training images and learn the detector. Looking at the solutions provided for these examples, we can see that they are either guided by a high level of supervision, such as considering the alignment to be close to the user annotation [6], [1], or guided by the bias of the dataset, such as considering the initial location to be in the center of the image in a dataset where most of the objects are already located close to the center of the images [12], [8]. In general, such weakly supervised learning problems are considered to be among the hardest problems in computer vision and to our knowledge no successful solution has been proposed for them. This is because, with no prior knowledge of how an object looks like and acknowledging the fact that different image descriptors such as HOG [2] and SIFT [9] are not accurate enough, finding the perfect correspondence between the images becomes a very challenging and computationally expensive problem.

In this paper, we propose a topological framework for the training of such latent variable models and address the problems of learning complexity and supervision in these models. In our framework, a sequence of sets of feature vectors is used to find an optimal decision boundary (Explained in §II). As we will discuss, these sets play an important role in our framework. While their size will directly relate to the computational complexity of the method, their content will determine to which solution the method will converge. We will argue that the strategy that is used for populating these sets plays a key role in the quality of the resulting decision boundary. Moreover, to address the supervision problem, we ask the questions, "Will the training framework still hold if no cue about the object is given to the model?", and if the model doesn't hold, "How can we formulate the desirable solution and automatically push the latent variables toward this solution?". To answer these questions, we formulate this as a weakly-supervised clustering problem and show that it can provide an efficient initialization for the latent variable models. Finally, to experimentally evaluate our method, we look at the problem of object classification with latent localization. This setup will provide us with an easy to evaluate framework which is very challenging to solve. Fig. 1, shows examples of this problem. In each image the blue box is the location that is initially considered to be the location of the object and the red box is the location found after the training is finished.

We organize this paper as following : In §II, we provide a proper definition of the problem and in §III, §IV and §V, we describe different strategies for solving this problem. In §VI, we experimentally evaluate these strategies and discuss the initialization problem. Finally, §VII concludes the paper.

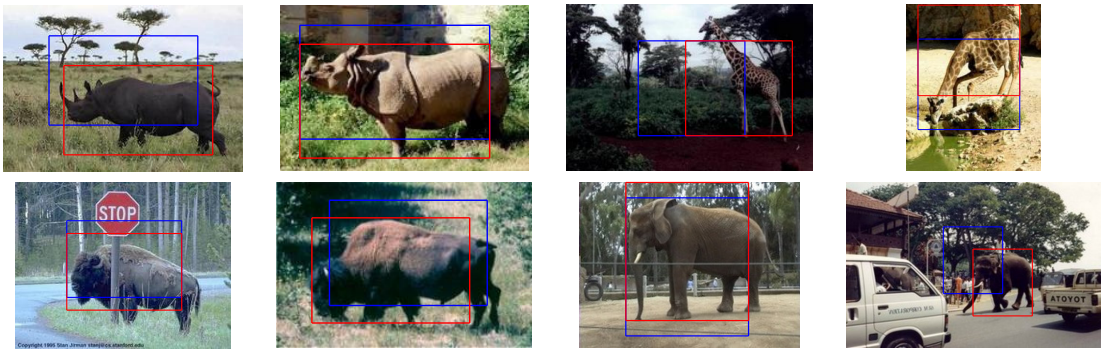

Fig. 1. This figure shows how the localization is done in our framework. Since the location of the object is not known, the blue box corresponds to the initial location of the object. During the training, the model allows the position of this box to change in order to find a better correspondence between the positive samples. The red bounding box is the location that corresponds to the object after the training.

II. Problem Definition

To formulate the problem, we assume that a dataset of labeled images $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ is provided, with $\mathbf{x}_i$ being the image and $y_i \in \{-1, 1\}$ being the binary label associated with it. For each image, there is a latent variable $h_i \in Z(\mathbf{x}_i)$ which localizes a fixed size bounding box. The content of this bounding box is encoded by the feature vector $\Phi(\mathbf{x}_i, h_i) \in \mathbb{R}^d$. In this problem, the task of the learning algorithm is to classify the images $\mathbf{x}_i$ according to the labeling y_i, while correctly localizing the object. If the accurate value of h_i is known for the training examples, then the problem becomes a standard detector training problem. However, with the assumption that this value is unknown, the training task becomes significantly more challenging. This is due to the fact that wrong fixation of this value can lead to training of inefficient models. The Latent SVM model (LSVM) [5] addresses this problem by minimizing the objective function

$$L_{\mathcal{D}}(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||2 + C\sum_{i=1}^{N} max(0, 1 - y_i f_{\mathbf{w}}(\mathbf{x}_n)), \qquad (1)$$

where

$$f_{\mathbf{w}}(\mathbf{x}_n) = \max_{z \in Z(\mathbf{x}_n)} \mathbf{w}^T \Phi(\mathbf{x}_n, z). \qquad (2)$$

This optimization is usually done by iterating between fixing the latent variables based on computed $\mathbf{w}$ and optimizing the model parameters $\mathbf{w}$ over the fixed problem. These iterations usually start by an initial fixation of the latent variables.

Looking at the problem of image classification with latent localization, our goal is to locate a feature vector that exists in all positive images and does not in negative images. Knowing about this feature vector allows us to localize the object (Localization) and score the image based on this localization (Classification). Since we do not have a prior knowledge of this feature vector in the training set, the training algorithm tries different fixations of latent variables until it converges to a proper solution with respect to the labeling of the images. In this paper, we look at this problem from a topological point of view [13], [10] and discuss the advantages of such perspective. Here, each image can be seen as a set of features, given by

$$C_n = \{\Phi(\mathbf{x}_n, h) : h \in Z(\mathbf{x}_n)\}. \qquad (3)$$

There are exactly N such sets, each corresponding to an image and any feature vector that can potentially be involved in the training exists in the set $X = \cup_{i=1}^N C_i \subset \mathbb{R}^d$. The sets $\mathcal{C} = \{C_1, \ldots, C_N\}$ can be seen as a cover for some topological space (X, τ) with $\tau \subset 2^X$.

In a topological space, two elements $\mathbf{a}, \mathbf{b} \in X$ are related iff there exists a set $A \in \tau$ such that $\mathbf{a}, \mathbf{b} \in A$. The elements of A does not necessarily have a geometrical relation with each other. This can be seen in the elements of each C_n. These elements are related since they come from the same image and not because there is a geometrical relation between them. In this paper, we will refer to a set $A \in \tau$ geometrical iff its elements have a geometrical relation with each other. As an example, $A \in \tau$ geometrical if it is populated by the elements of each C_n, with the property that they are optimal in a geometrical sense (Eq. 4 and Eq. 12). Using these definitions, we can formulate the problem in terms of constructing a sequence of geometrical sets $A_1, \ldots, A_M \in \tau$ with the property that the set A_{m+1} is populated based on the elements of A_m, by selecting feature vectors from the elements cover sets $C_1, \ldots, C_N$. Our aim here, is to construct a geometrical sequence in a manner that the set it converges to, successfully encodes the required geometrical properties that are necessary for localizing the object, while having a minimal size. Throughout this paper, we assume that the binary labeling of the elements of each A_m is known.

A good example of this procedure is the LSVM framework. The set A_m contains the one element from each image which corresponds to an initial fixation of the latent variables. A decision boundary $\mathbf{w}_m \in \mathbb{R}^d$ is trained over the elements of this set. Having this decision boundary, the set A_{m+1} is populated by the elements that maximize $\mathbf{w}_m$ from each C_n and every element from $A_p (p < m + 1)$ that is considered as *hard negative*. Such hard negative mining is considered to be essential for the convergence of the LSVM framework [5]. By following this procedure, we produce a sequence of sets $A_1, A_2, \cdots \in \tau$ with the property that if $i < j$ then $L_{\mathcal{D}}(\mathbf{w}_j) < L_{\mathcal{D}}(\mathbf{w}_i)$. Clearly, the sets $\{A_m\}_{m=1}^M$ control how the learning procedure proceeds. The content of these sets determines to which solution the method converges to and the size of these sets controls the complexity of the problem. Usually, training on larger sets requires more complex learning algorithms. While there are many computational advantages in keeping the size of these sets small, doing so can make the problem unstable and prevent the sequence from converging

to a solution.

The contributions of this paper are different strategies that can be used for constructing this sequence. We will discuss, how by taking different strategies it is possible to obtain an efficient and robust learning framework for such latent variable models. We will motivate these strategies from both theoretical and analytical perspectives. Finally, will show how our framework can be used to obtain a proper initialization for the training procedure.

III. Decision Boundary Strategy (DBS)

In this section, we discuss the conditions that are required for keeping the size of A_m sets low. Here, we assume that each A_m contains at least one feature vector from each image, which gives us the lower bound $|A_m| \geq N$. However, we will consider the worst case scenario, where for each m, $|A_m| = N$. Since $N \ll |X|$, there is a significant computational advantage in keeping the size of these sets low, which renders the process of making the intermediate models more simple. A downside of this assumption is the fact that at each iteration the model is trained on a very small fraction of the overall feature vectors and this can prevent the method from converging to a solution. To analyze this behaviour, we assume the set A_m has exactly N elements and each coming from a different cover set (image) C_n. Since the elements of A_m are labeled, we can train the linear classifier $\mathbf{w}_m$ over this set and let

$$A_{m+1} = \{\Psi(C_n, \mathbf{w}_m) : n \in \{1, \dots, N\}\}, \tag{4}$$

where

$$\Psi(C_n, \mathbf{w}_m) = \underset{\mathbf{f} \in C_n}{\arg\max}\ \mathbf{w}_m^T \mathbf{f}. \tag{5}$$

To determine the relation between A_m and A_{m+1}, let $\mathbf{a}_m^{(n)} \in A_m$ and $\mathbf{a}_{m+1}^{(n)} \in A_{m+1}$ be the elements coming from C_n and we state their relation through the following theorem.

Theorem 1. *Using the definitions above, if* $\mathbf{a}_m^{(n)} \in A_m$ *and* $\mathbf{a}_{m+1}^{(n)} \in A_{m+1}$ *then*

$$\mathbf{w}_m^T \mathbf{a}_m^{(n)} \leq \mathbf{w}_m^T \mathbf{a}_{m+1}^{(n)}. \tag{6}$$

Proof: Considering Eq. 5, the proof is straight forward. ■

This theorem states that the set A_{m+1} is populated by feature vectors with higher scores than the elements of A_m with respect to the decision boundary $\mathbf{w}_m$. In other words, by obtaining a feature vector with higher score, we are populating A_{m+1} with more positive-like feature vectors coming from both positive and negative images. This can result in a large difference between the elements of A_m and A_{m+1}, specially in the elements coming from negative images. This difference will cause $\mathbf{w}_{m+1}$ (trained over A_{m+1}) to focus on completely different attributes of the feature vectors compared to $\mathbf{w}_m$. This difference can also be motivated from a different perspective, namely that we have N labeled vectors and we are replacing a significant number of them by vectors that look more similar to the positive vectors. Clearly, the decision boundary of the original vectors is different from the one trained over the replaced vectors.

In general, we would like to have a relation between the sets A_m and A_{m+1} which translates to a relation between $\mathbf{w}_m$ and $\mathbf{w}_{m+1}$. As mentioned, in studies such as [5], this relation is encoded by adding the previously mined hard negatives to A_{m+1}. By doing so, they enforce a similarity between the decision boundaries $\mathbf{w}_m$ and $\mathbf{w}_{m+1}$ since most of the negative vectors used in their training is shared between them. Here, our goal is to keep $|A_m| = N$ and still encode such a relation between A_m and A_{m+1}. To do so, the following theorem states that if we add a correction term to the decision boundary of $\mathbf{w}_{m-1}$, based on the content of the set A_m, rather than using a newly trained decision boundary, then we can formulate such relation.

Theorem 2. *Assuming that the elements of* A_m *are selected by the decision boundary* $\mathbf{w}_{m-1}$ *and* $\mathbf{w}'_m$ *is the decision boundary trained over* A_m*. If we let*

$$\mathbf{w}_m = \mathbf{w}_{m-1} + \alpha_t(\mathbf{w}'_m - \mathbf{w}_{m-1}), \tag{7}$$

for some $\alpha \in [0, 1)$ *and select the elements of* A_{m+1} *using* $\mathbf{w}_m$ *then the following inequalities always hold:*

$$\mathbf{w}_m^T \mathbf{a}_{m+1}^{(n)} - \mathbf{w}_{m-1}^T \mathbf{a}_m^{(n)} \leq \alpha_t(\mathbf{w}'_m - \mathbf{w}_{m-1})^T \mathbf{a}_{m+1}^{(n)} \tag{8}$$

$$\mathbf{w}_m^T \mathbf{a}_{m+1}^{(n)} - \mathbf{w}_{m-1}^T \mathbf{a}_m^{(n)} \geq \alpha_t(\mathbf{w}'_m - \mathbf{w}_{m-1})^T \mathbf{a}_m^{(n)} \tag{9}$$

Proof: The proof can be found in the supplementary material. ■

It is clear from Eq. 7 that the smaller the value of α_t, the more similar $\mathbf{w}_m$ will be to $\mathbf{w}_{m-1}$. While many strategies can be revised for choosing this value, we will assume that this value will decrease as the sequence advances and we selected it to be $\alpha_m = \frac{1}{m}$. Here, in the early iterations, the value of α_m is rather large and this allows large changes in the elements A_m and after these iterations, as the value of α_w becomes smaller, the updates will simply fine tune the decision boundary. It should be mentioned that even though we are keeping the size of the sets A_m small, it is always guaranteed (Eq. 5) that it is populated with hardest negative feature vectors in the dataset with respect to the decision boundary. Because of this fact, the solution found by our method will be similar to the solution found by previously published methods.

IV. Euclidean Distance Strategy (EDS)

So far, we have discussed the problem in which each set in the sequence $\{A_m\}_{m=1}^M$ is populated, using a decision boundary obtained from the previous members of the sequence. In this section, we construct a sequence with its members populated based on euclidean distance rather than decision boundaries. We will show how using this strategy it is possible to find different feature vectors that are shared by all images and use them as an initialization seed to find better decision boundaries.

To formulate this problem, for a given set of images $\mathcal{C} = \{C_1, \dots, C_N\}$, we wish to find a point $\mathbf{p} \in \mathbb{R}^d$ such that a feature vector close to it exists in all images. To properly define this, we wish for this point to minimize the cost

$$\mathcal{L}_{\mathcal{C}}(\mathbf{p}) = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{p} - \Theta(C_n, \mathbf{p})\|^2, \tag{10}$$

where

$$\Theta(C_n, \mathbf{p}) = \arg\min_{\mathbf{f} \in C_n} \|\mathbf{p} - \mathbf{f}\|^2. \quad (11)$$

To find such a point, we construct a sequence of sets $\{B_m\}_{m=1}^{M} \subset \tau$ with the property that

$$B_{m+1} = \{\Theta(C_n, \mathbf{p}^{(m)}) : n \in \{1, \ldots, N\}\}, \quad (12)$$

where $\mathbf{p}^{(m)}$ is calculated as the mean of the elements in B_m. Without loss of generality, we can assume either $\mathbf{p}^{(0)}$ or B_1 are given. The following theorem shows that the cost function 10 decreases with the increase of m.

Theorem 3. *Given an imageset $\mathcal{C}$ and the points $\mathbf{p}^{(m)}$ and $\mathbf{p}^{(m+1)}$ defined as above, the following statement always hold:*

$$\mathcal{L}_{\mathcal{C}}(\mathbf{p}^{(m+1)}) \leq \mathcal{L}_{\mathcal{C}}(\mathbf{p}^{(m)}) \quad (13)$$

Proof: The proof can be found in the supplementary material. ■

This theorem shows that the sequence $\{B_m\}_{m=1}^{M}$ always converges to a point cluster in X with the property that each member of this cluster comes from a different image C_n. This cluster can then be used to initialize the method discussed in the previous section.

We can safely assume that a feature vector that is shared by the positive images, has a high likelihood of being the object of interest. To practically use this theorem, we will produce several of such clusters over the positive images and use cross validation to pick the most robust initialization. To guarantee that we converge to distinct clusters, for each B_m we will remove the elements that are closer to an already found cluster center than $\mathbf{p}^{(m-1)}$. In this case, the size B_m can be smaller than N. This formulation can be seen as a clustering method.

V. Mixed Strategies (MS)

In this section, we will discuss two heuristic mixed strategies for constructing the sequence. Here, not only do we wish to select the feature vectors based on their discriminative properties but also impose an euclidean similarity between the positive samples. To avoid confusion with the previous sections, these sequences will be denoted by $\{S_m\}_{i=1}^{M} \subset \tau$. Similar to §III, $|S_m| = N$ for all m and we assume that a decision boundary $\mathbf{w}_m$ is trained over S_m. For each image we define

$$C_n^m = \{\mathbf{f} \in C_n : \mathbf{w}_m^T \mathbf{f} > 0\}, \quad (14)$$

to be the set that contains every positively classified feature vector of C_n with respect to $\mathbf{w}_m$. We also define $\mathbf{p}^{(m)} \in \mathbb{R}^d$ to be the mean of the feature vectors within the set

$$P^m = \{\Psi(C_n, \mathbf{w}_m) : \mathbf{w}_m^T \Psi(C_n, \mathbf{w}_m) > 0, \forall n\}, \quad (15)$$

and $\mathbf{n}^{(m)} \in \mathbb{R}^d$ to be the mean of the elements within

$$N^m = \{\Psi(C_n, \mathbf{w}_m) : \mathbf{w}_m^T \Psi(C_n, \mathbf{w}_m) \leq 0, \forall n\}. \quad (16)$$

Using these definitions, we define

$$S_{m+1} = \{\Omega(C_n, \mathbf{w}_m) : n \in \{1, \ldots, N\}, \quad (17)$$

where

$$\Omega(C_n, \mathbf{w}_m) = \begin{cases} \Psi(C_n, \mathbf{w}_m) & C_n^m = \emptyset \\ \Theta(C_n^m, \mathbf{p}^{(m)}) & C_n^m \neq \emptyset \end{cases}. \quad (18)$$

In other words, this strategy picks the positively scored feature vector that is the closest to the average positive feature, and, when no positive features are detected, it simply picks the one with the highest score. We will refer to this strategy as "MS (1)". Similarly, we also consider another strategy in which we pick the positively scored feature vector which is both close to positive average and away from the negative average. The feature selection process of this strategy is defined similar to Eq. 18 with the difference that $\Theta(C_n^m, \mathbf{p}^{(m)})$ is replaced by

$$\Theta'(C_n, \mathbf{p}, \mathbf{n}) = \arg\max_{\mathbf{f} \in C_n} e^{-\|\mathbf{p}-\mathbf{f}\|^2} - e^{-\|\mathbf{n}-\mathbf{f}\|^2}. \quad (19)$$

We will refer to this strategy as "MS (2)".

VI. Experiments and Results

To experimentally analyze the properties of the discussed method and compare it with previously published methods, this paper uses the mammals dataset [7] which has been used to benchmark the methods in [8], [12] and follows their experimental setting. In these experiments, it is assumed that the objects have the same size and the main challenge is considered to be the localization of the object. To describe the image, we have used the HOG descriptor [2], [11]. Each experiment is repeated 10 times on random 50% splits of the dataset and the average performance is reported. While we use different strategies for training, the outcome of the training procedure is always a linear decision boundary for each class is used to fix the latent variables on the test images similar to [8], [12], [5]. We do not use any other information at the testing stage. In our model, the problem transforms into a fixed binary classification problem and LibLinear [3] is used for training this classifier. The timing of the codes is done on a single core of Intel Xeon 2.67GHz cpu using Matlab R2012b.

We divide the experiments into two parts. The first part, compares the performance of our formulation with the baseline [12] under the assumption that the latent variables are initially fixed at the center of each image. In the second part of the experiments, we discuss how having different initializations will affect the quality of the decision boundaries.

A. Center Initialization

As mentioned, in this section we follow the experimental setup of [8], [12] and assume that the latent variables are initially fixed at the center of each image. For this experiment, we employ the strategies DBS (§III), MS (1) and MS (2) (§V). Since the strategy EDS (§IV) is not discriminative, it does not apply to this experiment. In table I, we can see a comparison between the decision boundaries produced using different strategies and the baseline provided by [12]. As it can be seen, DBS is out performing the LSVM framework while using significantly lower number feature vectors for training. This difference can be caused by the fact that DBS uses less hard negatives, but a stepping mechanism to fine tune itself to find a better decision boundary. The strategy MS (1), which pushes positive features to look more similar, gives us a very small improvement over DBS with a higher standard deviation. Finally, MS (2) strategy, which pushes the positive features to be similar but away from negatives, produces the most precise decision boundary. It is interesting to see that the linear

Method	Classifier	Acc. (%)	Ref.	Learning Time (Sec.)
LSVM	Linear Classifier	75.07 ± 4.18	[12]	-
KLSVM	RBF Kernel	84.49 ± 3.63	[12]	-
DBS	Linear Classifier	80.15 ± 2.79	Section III	17.7
MS (1)	Linear Classifier	80.44 ± 4.43	Section V	26.30
MS (2)	Linear Classifier	$\mathbf{85.26 \pm 3.80}$	Section V	29.56

TABLE I. THIS TABLE GIVES A COMPARISON BETWEEN THE DIFFERENT STRATEGIES DISCUSSED IN THIS PAPER AND COMPARES THEM TO THE PREVIOUSLY PUBLISHED METHODS. IN ALL THESE EXAMPLES, WE HAVE ASSUMED THAT THE INITIAL BOUNDING BOXES ARE LOCATED AT THE CENTER OF THE IMAGES. THIS TABLE SHOWS HOW BY CHANGING THE STRATEGY OF POPULATING THE SETS OF THE SEQUENCE, WHILE KEEPING THE LEARNING FRAMEWORK UNCHANGED, IT IS POSSIBLE TO OBTAIN A SIGNIFICANTLY MORE ROBUST DECISION BOUNDARY. THE REPORTED TIME IS FOR TRAINING OF THE DECISION BOUNDARIES OF ALL SIX CLASSES.

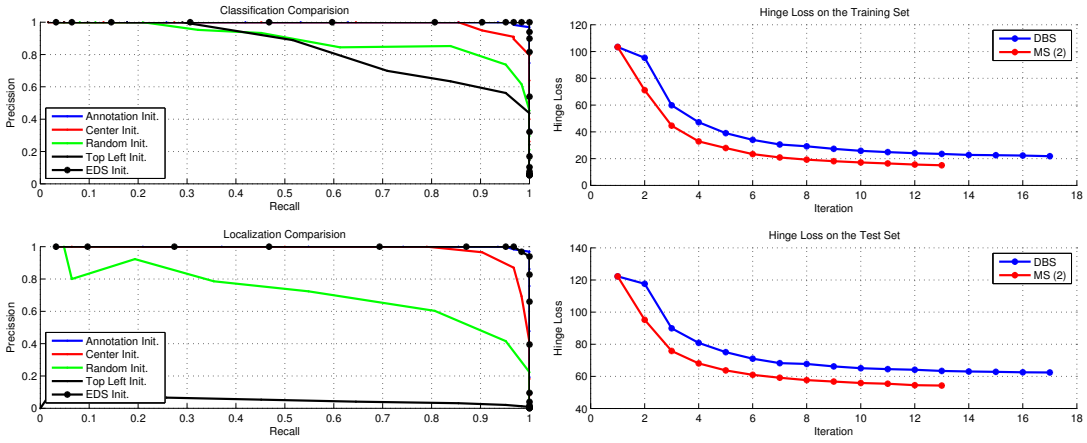

Fig. 2. **(Left)**This figure shows the results of the experiment where DBS was applied to train car detector on CALTECH-101 dataset. The plots in this figure show the effect of different initializations on the ability of the model to localize the object and classify the images. **(Right)** This figure shows how the hinge loss (Eq. 1) is minimized by our method on both training and testing sets. This visualization is based on only one of the classes of the dataset and a similar behaviour is observed for all the classes.

Init. Type	Acc. (%)	Time (Sec.)
Center	80.15 ± 2.79	-
Random	66.93 ± 3.56	-
Top Left	61.75 ± 3.06	-
Kmeans (10 Centers)	69.85 ± 2.15	5.0
EDS (10 Centers)	78.47 ± 3.91	0.6

TABLE II. COMPARISON BETWEEN THE CLASSIFICATION RATES OBTAINED USING DIFFERENT INITIALIZATION METHODS. THE LARGE DIFFERENCE BETWEEN THESE NUMBERS SHOWS THE SENSITIVITY OF THE LOCAL VARIABLE MODELS TO INITIALIZATION AND HOW IMPORTANT IS IT TO HAVE ROBUST METHODS FOR INITIALIZING THEM. IN THIS TABLE, EACH EXPERIMENT WAS REPEATED 10 TIMES AND THE AVERAGE PERFORMANCE IS REPORTED. THE TIMING PRESENTED IN THIS TABLE ONLY CORRESPONDS TO THE CALCULATION TIME OF THE CENTERS AND NOT THE TIME SPENT ON CROSS VALIDATION OF THE POINTS.

decision boundary produced by this strategy outperforms the KLSVM method with RBF kernel.

The results in table I show the importance of our discussions in this paper. While starting from the same initial fixation, different strategies taken to produce the sequence yield to significant difference in the quality of the decision boundary found by our method. In these experiments, we have demonstrated that with an effective strategy it is possible to produce a linear decision boundary that outperforms the previously published non-linear models.

B. Arbitrary Initialization

By taking a close look at the images in the mammals dataset [7] (Fig. 1), it is easy to spot that most of the objects are already located close to the center of the image and selecting the initial fixation to be at the center is an assumption that exploits the bias of this dataset. To show how this initialization aids the training process, table VI shows the performance of decision boundary found by DBS using different initializations. As it can be seen, there is large gap between the quality of the boundary when initialized at *center* with compared with when initialized at *top left* or a *random* location. Since none of these initialization are actually related to the content of the image, this gap only shows how we are exploiting the bias of the dataset when placing the initialization at the center.

To highlight this problem, we apply the same model to the *car* class of the CALTECH-101 [4] dataset. The goal here is to use the discussions of this paper to train a car model and benchmark it as an object detector. This dataset is interesting because, while the objects look rather similar across different images, each image contains many of the latent locations that do not overlap with the object. Here, we consider four different

initializations {*Annotation, Center, Random, Top Left*}. Fig. 2 (LEFT) shows the results of this experiment. Since this dataset is considered as one of the simplest available datasets, there is no surprise that by initializing at the annotation we obtain close to perfect classification and localization. Since this dataset also has the bias of most objects being located around the center of the image, we can see that the accuracy of the model is still high with center initialization. As it can be seen, once the initialization becomes more noisy, we see a large decrease in the accuracy of the model in both classification and localization. When initialized with top left location, the initialization has no overlap with the object. As we can see, the model trained under this condition completely fails to localize the object. The relatively higher classification performance of this initialization indicates there are other structures that are also shared by the positive images which do not over lap with the object.

To address the initialization problem, we use the EDS (§IV) strategy to produce 10 centers that are shared by the positive images. To do so, we initiate each sequence from a random feature vector coming from the training set and ensure that it doesn't converge to already found centers. Among these centers, we wish to pick the one that provides us with the most robust initialization. We score each center based on a 2-fold cross validation of DBS initialized using each center, on the training set. Since this process requires us to train the model many times, we employ the DBS method even though MS (2) has shown to produce better results (Table I). As a baseline, we also produce 10 cluster centers on the training images using the standard kmeans method and similarly pick the most robust initializer among them. As it can be seen in table VI, the initialization provided by EDS significantly outperforms the baselines and it is comparable with the center initialization. Similar results can be seen in Fig. 2(LEFT), that the model produced with EDS initialization performs similar to the model initialized at the annotation.

C. Convergence

As mentioned in §III, our method converges when a decision boundary reselects the set it was produced on. In practice, there are different ways of measuring this convergence. In this work we have considered the hinge loss to verify if the method has converged. This measure is both a proper approximation for our original definition of convergence and also relates our method to other methods which are based on decreasing the loss function given in Eq. 1. Fig. 2 (RIGHT), show how this loss is decreased by our method on both training and testing sets. In this figure, it should be noticed that MS (2) is more effective compared to the DBS in decreasing the loss and this is clearly reflected on the results given in Table I.

VII. Discussion and Conclusion

In this paper, we have addressed the problem of computational complexity and initialization issues in the training of latent variable models. The framework introduced in this paper uses a sequence of feature sets to converge to a solution. As we have demonstrated, these sets play a key role in our framework. The complexity can be controlled with the size of these sets and the quality of the decision boundary is directly related to the content of these sets. Our experiments show that it is possible to train robust decision boundaries while limiting the size of these sets. In this paper, we have also addressed the problem of initialization of the discussed latent variable models. A special formulation of our framework can locate cluster centers that are more likely to be the object of interest. To demonstrate this, we have shown that our method is capable of producing robust decision boundaries without taking advantage of dataset bias. While the experiments of this paper focus on the task of image classification with object localization, our method can be applied to any problem with a similar formulation. It should also be mentioned that the linear decision boundaries discussed in §III can be replaced with any discriminative model that resides in a linear algebraic space including kernel methods. Doing so will highly affect how the sets of the topological sequence are populated and the study of these effects is left to future works.

Acknowledgments

This work was supported by The Swedish Foundation for Strategic Research in the project Wearable Visual Information Systems".

References

[1] Hossein Azizpour and Ivan Laptev. Object Detection Using Strongly-Supervised Deformable Part Models. In *ECCV*, pages 836–849, 2012.

[2] Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection. In *CVPR (1)*, pages 886–893, 2005.

[3] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[4] Li Fei-Fei, R Fergus, and P Perona. Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on*, page 178, 2004.

[5] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object Detection with Discriminatively Trained Part-Based Models. *PAMI*, 32(9):1627–1645, 2010.

[6] Pedro F Felzenszwalb and Daniel P Huttenlocher. Pictorial Structures for Object Recognition. *IJCV*, 61(1):55–79, 2005.

[7] Geremy Heitz, Gal Elidan, Benjamin Packer, and Daphne Koller. Shape-Based Object Localization for Descriptive Classification. *International Journal of Computer Vision*, 84(1):40–62, March 2009.

[8] M Pawan Kumar, Benjamin Packer, and Daphne Koller. Self-Paced Learning for Latent Variable Models. In J Lafferty, C K I Williams, J Shawe-Taylor, R S Zemel, and A Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1189–1197. 2010.

[9] David G Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[10] B Mendelson. *Introduction to Topology*. Dover Books on Mathematics Series. Dover Publications, 1990.

[11] A Vedaldi and B Fulkerson. VLFeat: An Open and Portable Library of Computer Vision Algorithms. Technical report, 2008.

[12] Weilong Yang, Yang Wang, Arash Vahdat, and Greg Mori. Kernel Latent SVM for Visual Recognition. In P Bartlett, F C N Pereira, C J C Burges, L Bottou, and K Q Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 818–826. 2012.

[13] Afra J Zomorodian, M J Ablowitz, S H Davis, E J Hinch, A Iserles, J Ockendon, and P J Olver. *Topology for Computing*. Cambridge University Press, New York, NY, USA, 2005.