Bayesian Networks and the Imprecise Dirichlet Model applied to Recognition Problems

Cassio P. de Campos[1] and Qiang Ji[2]

[1] Dalle Molle Institute for Artificial Intelligence
Galleria 2, Manno-Lugano, Switzerland
[2] Rensselaer Polytechnic Institute
110 Eighth St., Troy, NY, USA
cassio@idsia.ch, jiq@rpi.edu

Abstract. This paper describes an Imprecise Dirichlet Model and the maximum entropy criterion to learn Bayesian network parameters under insufficient and incomplete data. The method is applied to two distinct recognition problems, namely, a facial action unit recognition and an activity recognition in video surveillance sequences. The model treats a wide range of constraints that can be specified by experts, and deals with incomplete data using an ad-hoc expectation-maximization procedure. It is also described how the same idea can be used to learn dynamic Bayesian networks. With synthetic data, we show that our proposal and widely used methods, such as the Bayesian maximum a posteriori, achieve similar accuracy. However, when real data come in place, our method performs better than the others, because it does not rely on a single prior distribution, which might be far from the *best* one.

1 Introduction

Bayesian Networks (BNs) encode joint probability distributions using a compact representation based on a directed acyclic graph where nodes are associated to random variables and conditional distributions are specified for variables given their parents in the graph. The adoption of BNs has increased in the past years. For instance, recent research in computer vision uses BNs for representing causal relationships in facial expression recognition, image segmentation, visual surveillance, activity understanding, among others [17, 21].

Accuracy of results relies on the quality of model parameters. Ideally, with enough data, it is possible to learn parameters by standard statistical methods like maximum likelihood (ML) or maximum a posteriori (MAP) estimations. However, learning reliable parameters may require a large amount of training data. In spite of that, approximate domain knowledge through constraints on parameters is available in many real applications and can improve estimations. We propose a framework for parameter learning that combines training data and domain knowledge in the form of constraints, and where imprecise priors are considered. We use the Imprecise Dirichlet Model (IDM) [18] to work with prior distributions so that we have a set of Dirichlet distributions on which we

apply the maximum entropy principle to obtain a final estimation. The imprecise priors may be viewed as a conservative choice when data are scarce to avoid overfitting. Furthermore, the proposed idea requires less hyper-parameters to be specified by the user (for instance, we do not need to define the prior of a maximum a posteriori estimation), which makes it reliable and adaptive, as shown in the experiments with real data. In our formulation, convex programming can be used, which quickly finds the global optimum solution. For incomplete data sets, a variant of the Expectation–Maximization method is used, where the expectation step is done as usual and the maximization is replaced with the new formulation. The methods are general and deal with Dynamic Bayesian Networks as well. Experiments with synthetic and real data from a facial action unit recognition and a human activity recognition captured with surveillance cameras show better results than ML and Bayesian MAP results, which are among the most used methods to learn such networks.

Previous work has either explored constraints together with a precise criterion, such as isotonic regression [10], closed-form solutions for the constrained ML estimation with complete data [16], constrained EM method with penalties [11], or has used an imprecise model, such as the naive and the tree-augmented credal classifiers [6, 7] or the imprecise decision trees [1]. Lukasiewicz [14] explores maximum entropy properties, but does not discuss a parameter learning procedure. de Campos and Cozman [4] work with constraints on priors and formulate the learning problem as a constrained optimization problem, but their formulation is restricted to complete data sets and uses a (somewhat slow) non-convex optimization procedure.

The paper is divided as follows. Section 2 introduces the notation and the problem of parameter learning. Maximum entropy and the Imprecise Dirichlet Model are presented, as well as constraints that can be used to guide the learning. Section 3 summaries the learning model and discusses the case of incomplete data. Section 4 presents experimental results and Section 5 concludes the paper.

2 BNs, Dynamic BNs and Parameter Learning

A BN can be defined as a triple $(\mathcal{G}, \mathcal{X}, \mathcal{P})$, where $\mathcal{G}$ is a directed acyclic graph with nodes associated to random variables $\mathcal{X} = \{X_1, \ldots, X_n\}$ (which we assume to be categorical), and $\mathcal{P}$ is a collection of parameters $p(x_{ik}|\pi_{ij})$, with $\sum_k p(x_{ik}|\pi_{ij}) = 1$, where $x_{ik} \in \Omega_{X_i}$ is a category or state of X_i and $\pi_{ij} \in \times_{Y \in \pi_i} \Omega_Y$ a complete instantiation of the parents π_i of X_i in $\mathcal{G}$ (j is viewed as an index for each parent configuration). In a BN every variable is conditionally independent of its non-descendants given its parents. The joint distribution is obtained by $p(x) = \prod_i p(x_{ik}|\pi_{ij})$, with $x \in \mathcal{X}$ and all x_{ik} and π_{ij} compatible with x.

We focus on parameter learning in a BN where the structure (i.e. the graph) is known. Given a data set D where each element is a sample of the BN variables, the goal is to find the most probable values for the whole parameter set $\mathcal{P}$. One way to quantify the result is by the log likelihood function $\log(p(D|\mathcal{P}))$. Assuming that samples are drawn independently from the underlying distribution, we

maximize $\log \prod_{ijk} p(x_{ik}|\pi_{ij})^{n_{ijk}}$, where n_{ijk} indicates how many elements of D contain both x_{ik} and π_{ij}. ML estimation has its optimum at $p(x_{ik}|\pi_{ij}) = \frac{n_{ijk}}{\sum_k n_{ijk}}$.

Dynamic Bayesian Networks (DBNs) can be viewed as two-slice temporal BNs, where at time zero, we have a standard BN just as described, and for slices 1 to T a *transitional* BN is defined over the same variables but nodes have parents on time t and/or time $t-1$. Conditional probability distributions for time $t > 0$ share the same parameters so that we can unroll the DBN to obtain the factorization $p(\mathcal{X}_{1:T}) = \prod_i p_0(X_i|\pi_i) \prod_{t=1}^{T} \prod_i p(X_i^t|\pi_i^t)$, where T is the number of slices, $p_0(\cdot)$ are conditional distributions of the initial BN and X_i^t, π_i^t represent the corresponding variables in time t. Learning parameters of DBNs is similar to the BN case, but we deal with counts n_{ijk} for both the initial BN and for the transitional BN. Thus counts are obtained from data sets with time sequences. In other words, we can reduce the learning problem in a DBN to the BN learning by carefully counting the frequencies on the data set. Therefore, all following discussion can be applied to both BNs and DBNs. We point out when additional care for DBNs is needed.

2.1 Constraints

When small amount of data is available, standard estimation methods may produce unreliable results. Constraints are available in many applications and may improve results. We describe here some constraints that may be accommodated in our learning procedure. In fact, such constraints can also be employed in the ML and the Bayesian MAP estimations, and we fully compare our method against these others (including their constrained versions). Constraints might be very effective, as we show for two computer vision problems (Section 4).

Let P be a sequence of parameters, α a corresponding sequence of constant numbers and β also a constant. A *linear relationship constraint* is defined as $\sum_{p(x_{ik}|\pi_{ij}) \in P} \alpha_{ijk} \cdot p(x_{ik}|\pi_{ij}) \leq \beta$, that is, any linear constraint over parameters can be expressed. Qualitative influences and synergies [19] are simple (but important) examples of linear constraints. Without quantitative statements, they allow us to encode that a given value for a variable makes more likely to observed another value in another variable, encoding an approximate domain knowledge. Other examples are sum of parameters, range, relationship, and ratio constraints [16], other types of influences and synergies, among many others. In fact, we have a very general assumption: constraints must define a convex parameter space, that is, any constraint in the form $h(P) \leq 0$, where h is convex, is allowed. Such flexibility helps us to properly describe our knowledge, while keeping the convexity assumption that guarantees a fast and global optimal algorithm. We have no restriction regarding the number of times a parameter appears in constraints or whether constraints involve distinct conditional distributions of the BN. They only need to be local to a node, otherwise they would violate the Markov condition of the BN. Although we do not use non-linear convex constraints in the experiments, they are also possible. To illustrate, suppose a *product relation-*

ship constraint defined as $\prod_{p(x_{ik}|\pi_{ij})\in P} p(x_{ik}|\pi_{ij}) \geq \beta$. Although non-convex, a simple log transformation makes it convex.

2.2 Imprecise Dirichlet Model

With the ML formulation, the idea is to fit parameters and data, even if the amount of data is very small. For example, with just a couple of samples, the estimation through ML will likely return undesired answers, as data are not representative of the actual distribution. Constraints applied to a ML formulation may help, but still the estimation will tend to the "incorrect" answer inside the space defined by the constraints. So only if the constraints are tight the performance will greatly improve. Another possible way not to obtain these unreliable estimations is to use a Bayesian approach, such as the MAP estimation, with a predefined prior. In this case the question is about the choice of the prior. If we can choose a good prior, such approach will lead to good estimations. However, in most cases the prior is hard to be selected and a *non-informative* prior is chosen, which may be far away from the correct distribution. The Imprecise Dirichlet Model (IDM) [18] alleviate such situations by introducing set-valued estimations instead of single point estimations. The idea is that parameters are more likely to be inside these sets, and so treating the whole set of estimations can lead to more reliable results.

In a Dirichlet model, the goal is to learn the parameters of multinomial distributions on $X_i|\pi_{ij}$ using training data and a Dirichlet prior as parametric model for $X_i|\pi_{ij}$, because of the conjugacy with the multinomial distribution [8]. A possible parametrization is $p(X_i|\pi_{ij}) \propto \prod_k p(x_{ik}|\pi_{ij})^{s\tau_{ijk}-1}$ for $s > 0$ and $\sum_k \tau_{ijk} = 1$, where the hyper-parameter s controls dispersion and hyper-parameters τ_{ijk} control location; s is interpreted as the *size* of a database encoding the same beliefs as the Dirichlet distribution. Using the IDM, s is fixed (usually between one and two [18]) but τ_{ijk} can freely vary between zero and one, so that our estimation lies in the interval

$$\frac{n_{ijk}}{s + \sum_k n_{ijk}} \leq p(x_{ik}|\pi_{ij}) \leq \frac{s + n_{ijk}}{s + \sum_k n_{ijk}}. \tag{1}$$

This roughly means that we are conservative with respect to the prior: instead of choosing a single prior, all possible priors (for a given s) are considered. We point out that we are using a local version of IDM, where the imprecision is (separately) considered for each local probability distribution that defines the Bayesian network. As mentioned, the advantage of this formulation is to avoid choosing the prior precisely as in the MAP estimation. Less hyper-parameters, more robust is the model and less sensitive to wrong user input choices. However, the outcome of IDM is a set of distributions. Next section describes and justifies maximum entropy as a way to select a single estimation from this set.

2.3 Maximum Entropy

The maximum entropy principle [12] can be used as a criterion to select a single conservative estimation from a set of distributions [1], in the sense that it

avoids drastic conclusions. For example, a binomial distribution without constraints has the uniform distribution as the entropy maximizer. Furthermore, the distribution of maximum entropy from a set of distributions learned with IDM agrees (in the limit) with relative frequencies [18]. So, our goal is to have a learning model that achieves better solution for small amount of data, but which still tends to frequencies (as it should) when enough data are available. Note that the application of maximum entropy goes towards the opposite of ML, so it might seem at first contradictory, since we want to fit model and data. However, maximum entropy is employed only inside the learned IDM, which is responsible for the fitness (but considering all possible priors), while the idea of picking the distribution of maximum entropy (inside IDM) avoids overfitting by selecting the least fitting model among the IDM distributions. Thus, a possible objective is $\max_{\mathcal{P}} -\sum_{ijk} p(x_{ik}|\pi_{ij}) \log p(x_{ik}|\pi_{ij})$, which is put together with Equation (1), simplex constraints to ensure that answers are probability distributions and convex constraints defined in Section 2.1. The set of all such restrictions is denoted as $\mathcal{C}$. This formulation is based on the *local* maximum entropy criterion, that is, maximization is performed for each local conditional probability distribution in the network. Another approach is the Sequential Maximum Entropy [14]. However, such more sophisticated idea cannot (at least in a straightforward way) handle general constraints among parameters of distinct (yet local) distributions (even simple qualitative influences [19] relate parameters of distinct distributions). In Section 4 we present empirical results that support the choice of local maximum entropy.

3 The Learning algorithm

In this section we summarize our formulation to solve the learning problem. For complete data, the idea is simple: all pieces described so far (constraints, IDM, and maximum entropy) lead to a convex optimization program. Just as likelihood, entropy is concave, so we have a maximization of a concave function subject to a collection $\mathcal{C}$ of convex constraints on parameters and the intervals of IDM of Equation (1). The important technical detail that is worth mentioning is that we use some auxiliary optimization variables to deal with the following situation: constraints of $\mathcal{C}$ defined by the expert can force parameters to lie outside the interval of Equation (1) imposed by the IDM, that is, in this case the problem would be unfeasible. However, we assume that expert's constraints are always correct and shall be included in the model only if the expert is completely sure about them (e.g. physical and physiological aspects, logical rules, domain scope, etc). Because of that, they receive more importance than Equation (1). To quantify this importance, we adopt an approach where the IDM interval must be satisfied as much as possible, while constraints of $\mathcal{C}$ must be always satisfied. The role of Equation (1) is to bring the estimation close to frequencies of parameters in the data set: (i) if Equation (1) and expert's constraints are not disjoint, then there are solutions (in the intersection if these sets) that satisfy all of them; one of them will be selected by entropy; (ii) if IDM intervals and

expert's constraints are disjoint, we choose to first satisfy expert's constraints, but preferring estimations that are as close to the IDM intervals as possible. We leave for future analysis other ways to put together IDM and expert's constraints.

In this formulation, as the data set is smaller, as the result is more conservative, because of the entropy maximization and the wider intervals of Equation(1); as the data set is larger, as the result is closer to the ML estimation, because the interval shrinks: $\frac{s+n_{ijk}}{s+\sum_k n_{ijk}} - \frac{n_{ijk}}{s+\sum_k n_{ijk}} = \frac{s}{s+\sum_k n_{ijk}} \to 0$ as the values n_{ijk} increase (s is fixed). Hence, $p(x_{ik}|\pi_{ij})$ becomes closer to $\frac{n_{ijk}}{\sum_k n_{ijk}}$. Thus, the formulation is (automatically) more careful with scarce data and more aggressive towards the ML estimation with abundant data. We will see in the experiments with real data that this formulation provides a better trade-off than ML and than MAP. On the computational side, we have the challenge of solving the convex optimization. For example, we can use specialized interior point solvers or even some general optimization ideas, because convex programming has the attractive property that any local optimum is also a global optimum. Furthermore, such global optimum can be found in polynomial time in the size of input [3] (almost as fast as linear programming). Note that the input size here is small, as the problem can be solved for each node separately.

In the remaining of this section we discuss how to deal with incomplete data. Both the log-likelihood function and our formulation become non-convex, because the counts n_{ijk} from the data set are not precisely known. A common method to overcome this situation is the Expectation-Maximization (EM) algorithm [9], which starts from some initial guess, and then iteratively takes two types of steps (E-steps and M-steps) to get a local maximum. Particularly for discrete nodes, E-step computes the expected counts using the parametrization of the previous step, and M-step estimates new parameters by maximizing the likelihood function, given the counts from E-step, just like if a complete data set was in place. We can perform the same idea in our formulation. The E-step computes expected counts as usual, and the M-step is replaced by our formulation, with the constraints from $\mathcal{C}$ and from IDM. We stop when there is no possible improvement. Because of the convexity of the parameter space and the global optimizer, it suffices to include an extra linear constraint on $\mathcal{C}$ that forces the optimizer of the M-step to pick always an improving solution in case there is one, and thus the algorithm converges in the very same way as the EM. We cannot guarantee that it converges towards a local optimum, as the original EM also does not [20], but local optima are empirically verified in most situations. About time complexity, the time spent with the new idea is dominated by the E-step (which needs to perform queries in the network), and thus it is roughly as fast as the original EM version (which needs to run the same E-step).

This same Expectation–Maximization idea can be straightforward applied to DBNs. Note that the modified EM just described has a new M-step, but keeps the E-step unchanged. DBNs require the inference procedure that evaluates the expected counts to be adapted. As our formulation does not affect the E-step, we can directly apply any usual inference method of DBNs. We have used the Online Junction Tree Inference Algorithm [15] to obtain the expected counts,

and then we treat the initial and transitional parts separately as if a complete data set was in place. This is employed in Section 4.2 for activity recognition.

4 Experiments

We apply the idea to both synthetic and real data of distinct computer vision problems with the aim of showing its generality and applicability to other problems. Figure 1 presents results for synthetic data of ML, Bayesian MAP with Dirichlet prior, local maximum entropy and IDM, and a formulation using sequential instead of local maximum entropy. The bars are average Kullback–Leibler (KL) divergences for 20 runs of constrained ML (first bar of each series), constrained MAP (second bar), constrained local maximum entropy and IDM (third bar), and constrained sequential maximum entropy and IDM (last bar of each series). The runs use random networks (random graphs with up to four states per variable), constraints and data. The size of the networks and data are presented in the figure's labels. Networks have 10, 20 and 40 nodes and data sets have 10, 100 and 500 samples. The same set of constraints and data were applied to each method in each iteration. We use randomly generated constraints in number equal to the number of local distributions in the network. We employ a constrained MAP formulation that uses a prior where τ is defined uniformly. MAP, local entropy with IDM and sequential entropy with IDM achieve similar results, and they are better than likelihood estimations with respect to KL divergences. For complete data, in some test cases MAP is slightly better than maximum entropy, while in others it is slightly worse. There is an advantage of the entropy with IDM against MAP, which relates to the amount of information the user must provide. For random networks, the choice of uniform τ as prior is reasonable and achieves good results. On the other hand, maximum entropy with IDM does not depend on a single prior but considers all possible priors, so achieving similar results as MAP (which has the correct prior) is a positive attribute, as maximum entropy with IDM requires less information as input and is clearly more adaptive (e.g. in real data domains).

4.1 Facial Action Unit Recognition

We now consider the problem of recognizing facial action units from real image data [13]. Based on the Facial Action Coding System, facial behaviors can be decomposed into a set of Action Units (denoted as AUs), which are related to contractions of specific sets of facial muscles. We work with recurrent 14 AUs.[1] Some AUs happen together to show a meaningful facial expression: AU_6 (cheek raiser) tends to occur together with AU_{12} (lip corner puller) when someone is smiling. On the other hand, some AUs may be mutually exclusive: AU_{25}

[1] AU_1 (inner brow raiser), AU_2 (outer brow raiser), AU_4 (brow lowerer), AU_5 (upper lid raiser), AU_6 (cheek raiser and lid compressor), AU_7 (lid tightener), AU_9 (nose wrinkler), AU_{12} (lip corner puller), AU_{15} (lip corner depressor), AU_{17} (chin raiser), AU_{23} (lip tightener), AU_{24} (lip presser), AU_{25} (lips part), and AU_{27} (mouth stretch).

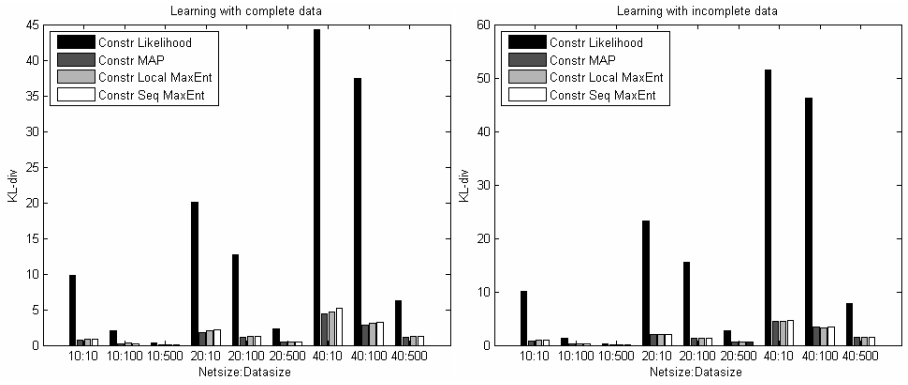

Fig. 1. Comparison between methods for synthetic random generated models using the KL divergence from the correct model. ML, even with constraints, is considerably less accurate than others. The right graph excludes constrained ML to clarify that differences are small among the other methods.

(lips part) never happens simultaneously with AU_{24} (lip presser) since they are activated by the same muscles but with opposite motions.

A BN with 14 hidden nodes is employed, which has already demonstrated good performance in the literature [5, 17]. Each node is associated to an AU with two states: activated and deactivated. Figure 2 depicts the structure of the BN. Note that every link between nodes has a sign, which is provided by a domain expert. Signs indicate whether there is positive or negative qualitative influence between AUs. For example, it is difficult to do AU_2 (outer brow raiser) alone without performing AU_1 (inner brow raiser), but we can do AU_1 without AU_2. The constraints are mainly based on physiological aspects, e.g. *mouth stretch* increases the chance of *lips apart*, and it decreases the chance of *cheek raiser and lid compressor* and *lip presser*. *Cheek raiser and lid compressor* increases the chance of *lip corner puller*. *Upper lid raiser* increases the chance of *inner brow raiser* and decreases the chance of *nose wrinkler*. *Nose wrinkler* increases the chance of *brow lowerer* and *lid tightener*. *Lip tightener* increases the chance of *lip presser*. We note that constraints are not tuned, but created by an expert.

Furthermore, 14 measurement nodes (unshaded in Figure 2, one for each AU) represent results derived from computer vision techniques. Links between AU and measurement nodes represent uncertainties in classifications. To obtain the measurement for each AU, first the face and eyes are detected in the images, and the face region is extracted and normalized based on the detected eye positions. Then each AU is detected individually by a two-class AdaBoost classifier with Gabor wavelet features [2]. The output is employed as the AU measurement in the BN model. For each measurement node, a domain expert provides ranges (usually tight) for $p(O_i|\mathrm{AU}_i)$, which represent accuracy of classifiers.

We use 8000 images from Cohn and Kanade's DFAT-504 [13]: 20% are separated for testing and 80% for training (although just part of it is used at each

time). We work with two data sets: one generated from computer vision measurements (used as evidence for testing) and one from human labeling (used for training), where uncertain labels are missing (data are incomplete). In Figure 3 we consider training data with 10, 100, 200 and 500 samples, randomly selected 20 times from the training set (results are averaged; standard deviation is below 4 pp. in all cases, mostly below 2 pp.; the graph is in log-scale). We can see that the constrained maximum entropy is superior in all cases, and that such superiority begins to vanish as more data are used. Although our main goal is to compare the learning procedures against each other rather than surpassing the state-of-the-art methods, we obtain an overall recognition rate (percentage of correctly classified cases) of about 94% using just 1000 training samples, which is comparable to state-of-the-art results [17].

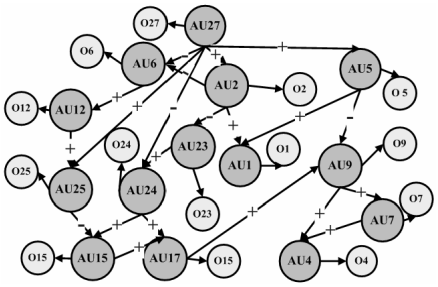

Fig. 2. Network for the AU recognition problem.

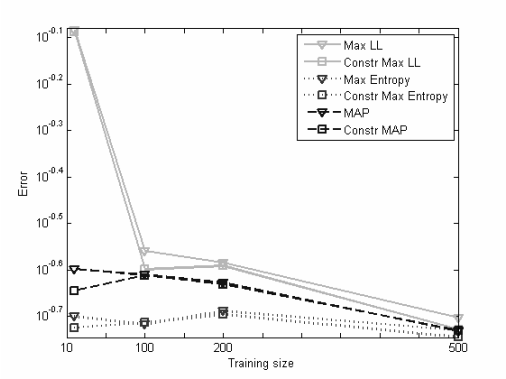

Fig. 3. AU recognition error rate (in log scale) on some BN parameter learning approaches.

4.2 Activity recognition

We also evaluate our approach using DBNs on a human activity recognition data set captured with a surveillance camera at a parking lot. The data set contains 110 sequences and we want to classify 7 possible activities: walking, running, leaving car, entering car, bending down, throwing and looking around. We first train DBNs for each activity and classify the activity according to the DBN with best fit for each test case. Their structure is shown in Figure 4.2, with three hidden nodes for position (Y), shape (S) and speed (V), and corresponding observation nodes. Each hidden node has two states. Temporal links exist from the time t to time $t+1$ for each node. Furthermore, the temporal link between V^t and Y^{t+1} encodes the dynamic relationship between speed and position.

The measurements of the observation nodes are obtained from the motion detection results. We first perform background subtraction to detect the motion

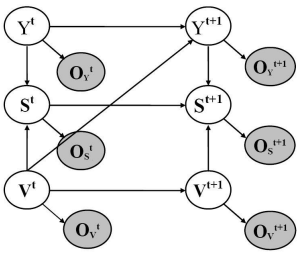

Fig. 4. DBN structure for human activity recognition.

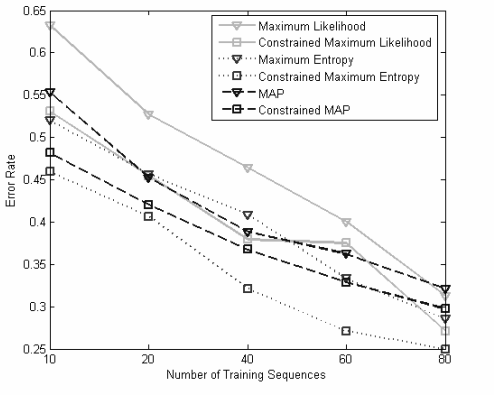

Fig. 5. Activity recognition results (error rate) using many DBN parameter learning approaches.

blob of the object. Position O_Y is measured as the distance to the car with 6 discrete values. Speed O_V is evaluated as the change of the blob center in pixels, which is discretized to 6 states. Shape measurements O_S are clustered in 4 features based on the aspect ratio of the bounding rectangle, filling ratio (the area of the blob with respect to the area of its bounding rectangle) and two first order moment features [21].

Given general constraints about smoothness, dynamics and physical attributes of the environment, we create a set of constraints on model parameters. Such constraints are applied to transitional probabilities of all activities, as they are general enough to be common to all activities. We note that constraints are acquired from experts and are not tuned to this specific data. We omit specific details about them because it would drive us far from the goal of this paper.

We randomly choose part of the activity sequences as training data and use the rest for testing. We compare ML, maximum a posteriori (MAP), and local maximum entropy, using both the unconstrained and constrained versions. We point out that other learning techniques do exist, such as regularized ML. Because we are discussing about a Dirichlet model and because ML with some regularization produces similar results as MAP, we have chosen to work with the latter instead. The training sets have 10, 20, 40, 60 and 80 sequences. For each size, we perform the test 10 times and the average recognition error is presented in Figure 4.2. It is worth noting that: (i) the use of constraints significantly decreases the error rate, as may be noted in Figure 4.2 by taking the curves two by two and analyzing constrained and respective unconstrained versions; (ii) when large amount of data is available, all methods tend to similar results. We can see that ML and MAP are already much closer when 80 training sequences are considered, and the same starts to happen with IDM; (iii) while results with synthetic data are not conclusive when comparing MAP and IDM with maximum entropy (the prior applied to MAP was enough there, and MAP was

even slightly better), here constrained maximum entropy outperforms all others, for all test cases. This is probably because all priors are considered by the IDM, while MAP uses a single one. MAP might achieve better results, but that would strongly depend on the quality of the prior chosen by the expert.

5 Conclusion

This paper presents a framework for parameter learning using the Imprecise Dirichlet Model and its application to two recognition problems. Domain knowledge in the form of constraints is exploited to improve accuracy. To select a single (conservative) distribution from the Imprecise Dirichlet Model, maximum entropy is used. The framework is very fast and guarantees to find the global optimal solution for complete data. For incomplete data, we propose to use an adapted version of the Expectation–Maximization method. Empirical results with synthetic data support the method. Both with synthetic and real data, we have compared the method against widely used maximum likelihood and Bayesian maximum a posteriori estimations, with and without constraints. We point out that the described method has the advantage of not requiring the specification of a single prior, as it is done by the MAP estimation. MAP with a good prior may obtain good results, but that strongly relies on the quality of such prior. Selecting a single prior is not an easy task when dealing with real data, and the uniform is used in most cases, which may lead to inferior results. Using the data to select the prior might overfit the model, resulting in unreliable results that are only applicable to the specific contexts and data.

Empirical results with real data are treated for two computer vision problems: a facial action unit recognition problem and an activity recognition problem in video sequences. The constraints that are used in each problem are described, which translate the domain knowledge into a mathematical formulation. We note that the constraints were defined by an expert once and were not tuned. As expected, results with constraints are superior than those when constraints are not used. Furthermore, the Imprecise Dirichlet Model shows better accuracy in all scenarios with real data. Specifically, the single prior of MAP estimation employed while using real data does not achieve as good results as with synthetic data, and the IDM (which considers all possible priors) outperforms the other methods. In summary, we point out that (i) constraints are very helpful when scarce data are available, which is a common situation in computer vision problems; (ii) widely used methods such as maximum likelihood and Bayesian MAP estimators, which are the most common ideas to learn Bayesian network parameters, are defeated by the IDM plus maximum entropy when dealing with real data and constraints, even though they performed well for synthetic data.

Acknowledgments This work is supported in part by the grant W911NF-06-1-0331 from the U.S. Army Research Office, and by the *Computational Life Sciences (CLS)* program phase II, canton Ticino, Switzerland.

References

1. Abellan, J., Moral, S.: Maximum of entropy for credal sets. Int. Journal Uncertain. Fuzziness Knowl.-Based Systems 11(5), 587–597 (2003)
2. Bartlett, M.S., Littlewort, G.C., Frank, M.G., Lainscsek, C., Fasel, I.R., Movellan, J.R.: Automatic recognition of facial actions in spontaneous expressions. Journal of Multimedia 1(6), 22–35, (2006)
3. Ben-Tal, A., Nemirovski, A.: Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications. MPS/SIAM Series on Optimization, SIAM (2001)
4. de Campos, C.P., Cozman, F.G.: Belief updating and learning in semi-qualitative probabilistic networks. In: Conf. on Uncertainty in Artificial Intelligence. pp. 153–160 (2005)
5. de Campos, C.P., Tong, Y., Ji, Q.: Constrainted maximum likelihood bayesian network for facial expression recognition. In: European Conf. on Computer Vision. LNCS vol. 5304, pp. 168–181 (2008)
6. Corani, G., de Campos, C.P.: A tree augmented classifier based on extreme imprecise dirichlet model. Int. Journal Approx. Reasoning 51, 1053–1068 (2010)
7. Corani, G., Zaffalon, M.: Learning reliable classifiers from small or incomplete data sets: the naive credal classifier 2. Journal of Machine Learning Research 9, 581–621 (2008)
8. DeGroot, M.: Optimal Statistical Decisions. McGraw-Hill, New York (1970)
9. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Stat. Society B 39(1), 1–38 (1977)
10. Feelders, A., van der Gaag, L.C.: Learning Bayesian network parameters under order constraints. Int. Journal of Approximate Reasoning 42(1-2), 37–53 (2006)
11. Graca, J., Ganchev, K., Taskar, B.: Expectation maximization and posterior constraints. In: NIPS. pp. 569–576. MIT Press, Cambridge, MA (2007)
12. Jaynes, E.T.: Information theory and statistical mechanics. Physical Review 106, 620–630 (1957)
13. Kanade, T., Cohn, J.F., Tian, Y.: Comprehensive database for facial expression analysis. In: Proceedings of the 4th IEEE Int. Conf. on Automatic Face and Gesture Recognition. pp. 46–53 (2000)
14. Lukasiewicz, T.: Credal networks under maximum entropy. In: Conf. on Uncertainty in Artificial Intelligence. pp. 363–370 (2000)
15. Murphy, K.P.: Dynamic Bayesian Networks: Representation, Inference and Learning. Ph.D. thesis, Univ. of California, Berkeley (2002)
16. Niculescu, R.S., Mitchell, T.M., Rao, R.B.: A theoretical framework for learning bayesian networks with parameter inequality constraints. In: Int. Joint Conf. on Artificial Intelligence. pp. 155–160 (2007)
17. Tong, Y., Liao, W., Ji, Q.: Facial action unit recognition by exploiting their dynamic and semantic relationships. IEEE Trans. on Pattern Analysis and Machine Intelligence pp. 1683–1699 (2007)
18. Walley, P.: Statistical Reasoning with Imprecise Probabilities. Chapman and Hall, London (1991)
19. Wellman, M.P.: Fundamental concepts of qualitative probabilistic networks. Artificial Intelligence 44(3), 257–303 (1990)
20. Wu, C.F.J.: On the convergence properties of the EM algorithm. The Annals of Statistics 11(1), 95–103 (1983)
21. Xiang, T., Gong, S.: Beyond tracking: Modelling activity and understanding behaviour. Int. Journal of Computer Vision 67(1), 21–51 (2006)