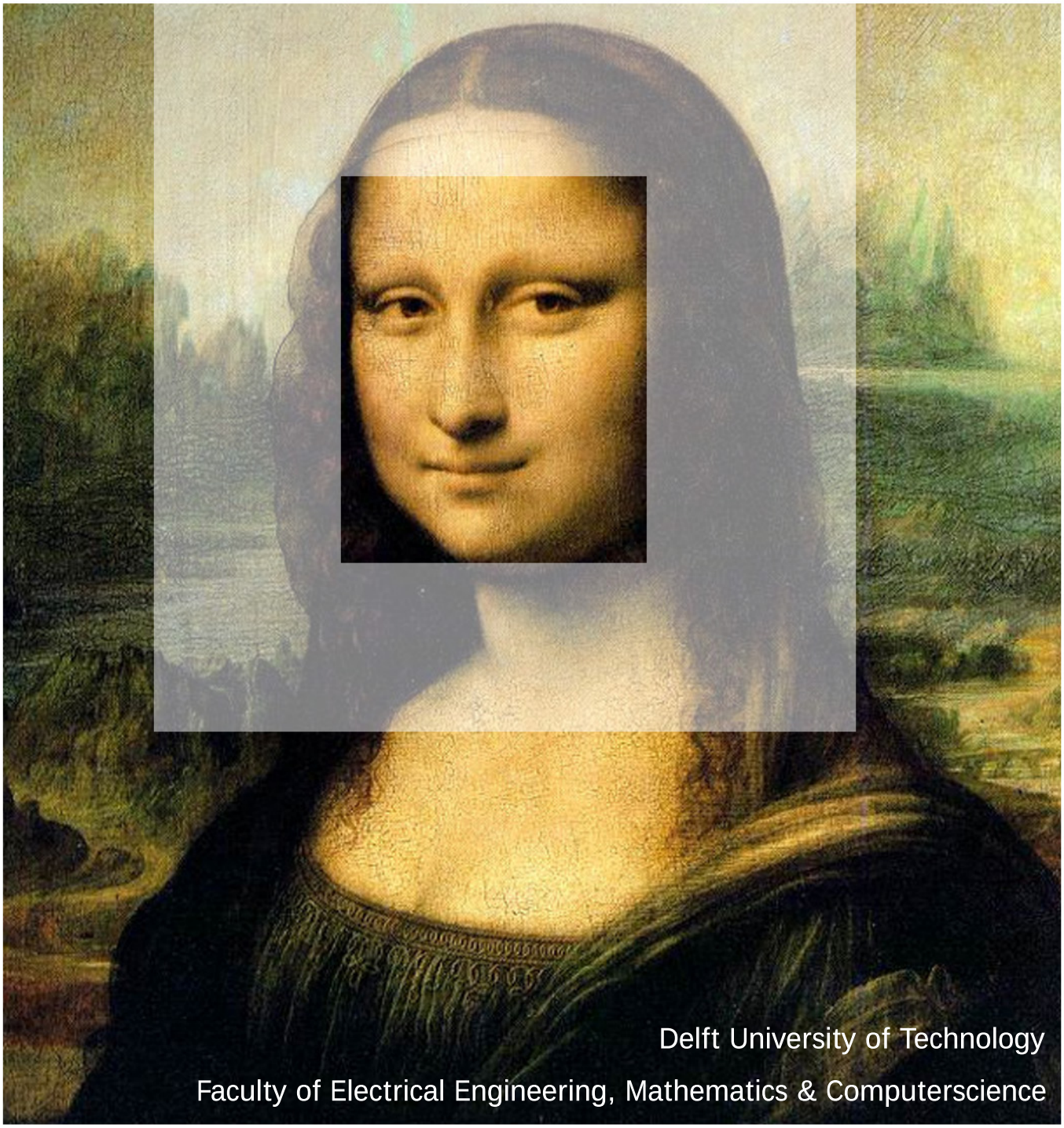

Bayesian Networks

Applied to

Facial Expression Recognition

TU Delft

Abstract

This literature survey gives a theoretical overview of Bayesian networks (BN) and discusses the application of Bayesian networks to the problem of facial expression recognition (FER). A Bayesian network is a graphical representation of a probabilistic system that models the full joint conditional probability distribution of an arbitrary problem. Facial expression recognition aims to determine the emotional state of a subject by analysis of the facial features. Different structures for Bayesian networks are reviewed such as Naïve Bayesian networks (NB), Tree-Augmented Naïve Bayesian networks (TAN) and Hybrid Bayes. Bayes is compared to other methods such as Support Vector Machines (SVM), Relevance Vector Machines (RVM) and AdaBoost. Furthermore, the small sample case is discussed with regard to the Bayesian approach, which is particularly vulnerable to a lack of sufficient training data. We discuss several solutions for the small sample case and some final recommendations are made for future research.

Summary of Contents

1 Introduction

The smile of the Mona Lisa is perhaps the most illustrious facial expression known to man. Although there are numerous theories as to the cause of the smile, ranging from the 'highway blues' theory by Bob Dylan [1] to the alleged self-portrait theory defended by Dr. L. Schwartz [2], there seems to be no controversy over Mona Lisa's portrayed facial expression. This is quite interesting, since we may safely assume that there is no one alive today who has actually known Mona, that is to say who has extensive knowledge of her facial features, it seems particular that almost every person would recognize the slight smile upon her face. Apparently, we humans are uniquely qualified to recognize certain facial expressions and attribute some emotional state to the person portraying the observed expression. Of course we may argue that pattern recognition technology can be used to determine key points on non-rigid objects such as the human face. The relative key points can be used to systematically analyze observed facial expressions. And it has been argued that we can map facial expressions directly to emotional states. Yet when we want to use modern technology to simulate recognition in an automated way, we encounter a true challenge.

1.1 Human-Computer Communication

Humans communicate. We do so in a lot of different ways, often using more than one mode of communication at the time. We converse with words, gestures and facial expressions, none of which is insignificant. Although humans are capable to communicate with only written words, we often need more than an exchange of words to achieve a robust level of communication. Mere words are often context dependent, so we need some indication of their context, meaning we need to asses the current mental state of the person that is trying to communicate with us. One way humans portray their mental state is by using facial expressions. When in conversation we often use our face to clarify our words. For instance, if a person with a happy face would tell us that he had lost his wallet, we would be inclined to think that this person was telling a joke, while if he that person would have had a sad face, we would have been inclined to believe his or her statement to be true.

Figure 1.1: Sony's AIBO

The field of Human-Computer Interaction strives towards a comparable level of communication. A computer that could interact with humans through facial expressions would advance human-computer interface towards a standard comparable to human-human interaction. Today computers can still be seen as 'emotionally challenged' as they fail to recognize the emotions of the humans with whom they interact. However, if it were possible in some way for computers to become emotionally sensitive, this would greatly enrich the possible communication between humans and computers. Obviously, non-verbal communication play a big part in everyday life. Not only the expressions on faces are important, but also gestures such as the wink of an eye or a stuck out tongue are a form of non-verbal communication. Facial expression recognition can be applied in number of different contexts. It can easily be imagined that a intelligent agent designed to be aware of its environment, for example Sony's AIBO, will be greatly improved if it can say something about the emotional state of the subject with whom it is interacting. For instance it might try to cheer up a person if it detects sadness, or increase its awareness if it senses fear. Another area of application

might face recognition in order to identify certain persons, in which case reverse mapping can be useful to match the person portraying a particular facial expression in a video stream back to the expression on the queried image of a neutral face. In a nutshell: facial expression recognition provides us with a way to improve the quality of communication between humans and machines.

1.2 Applications for automated facial expression recognition

Facial expression classification has a number of applications. As discussed in the previous section, most of them are focused at improving the communication between humans and machines. The previous section already gave the example of Sony's AIBO as a possible application for expression recognition. Another application might be an automated agent like Microsoft's 'Office Assistant' which decides whether or not to intervene with a helpful suggestion, based on the current emotional state of the user. When an observation of the user indicates severe frustration (not uncommon) the agent could be conditioned to provide help immediately. On the other hand, if the agent notices an increase in the frustration level of the user after intervention, it could be conditioned to be less likely to provide help in the future. Such an agent is suggested in [20].

The existence of emoticons in chat rooms can be largely explained by the lack of context in the conversations. For example, It is hard to discern sincerity from sarcasm (context) when written words are the only channel of communication. Of course, when two humans converse in a chat room, this problem can be easily solved by a set of webcams. However, when a human is conversing with a computer, facial expression recognition provides the means to determine context or can even provide a communication channel in itself. In the movie '2001: a space odyssey' the artificial intelligence HAL discovers the plot to shut him down, by lip-reading one of the astronauts. This idea might have seem farfetched when the movie premiered in 1968, but it seems a whole lot more feasible now.

One might also imagine a system that monitors people for certain expressions. The paper [19] describes a system that recognizes violent behavior in a group of people by monitoring their movement patterns. The problem is that this is not a pre-emptive approach, as the violence is detected only after it has occurred and persisted for a certain amount of time. Facial expression recognition might be used to recognize violence in a pre-emptive manner by detecting anger or fear on the observed faces, possibly enabling the police to prevent the situation from escalating. A similar application is detecting pain by analyzing facial expression (Pantic, Rothkrantz 2003). Patients in hospitals are monitored by keeping track of their blood pressure or hart beat-rate, but these are not infallible indicators of pain.

1.3 Topics in automated facial expression recognition

To get an idea of the pivotal topics in facial expression recognition, we need to know the problems commonly encountered and how facial expression recognition is applied. In general, automated facial expression recognition can be divided into three separate processes: detecting the face, extracting the salient features and classifying the expression. Each of these steps comes with its own set of problems.

The first step is finding the face in an image or video stream. When looking at facial expression recognition models, the input can consist of either a static images or video streams. The main difference is the way we locate the face in the input image/stream. Furthermore, video streams also give temporal information. For instance, if a face expresses joy in a particular frame, the probability that it will express joy in the consecutive frame is high. Intuitively this makes sense, but how to incorporate this into a model is another question entirely. The early facial expression recognition techniques used only static images, but when the interest shifted from an analytical point of view to a more practical point of view in the 1970s, recognition from video streams became more popular.

When facial expression recognition is applied to static images, the face can be located by determining the location of the face as a whole (holistic approach) or by identifying the face by looking for certain facial features such as the eyes (analytic approach). Locating the face in a video stream is somewhat more challenging. A common method is to first find all moving blobs in a video stream. Consecutively all blobs that might represent faces are identified. Once the face has been found, the model keeps track of the blob representing the face by comparing pixel values. This technique is known as *tracking*. An additional problem is that

some parts of the video stream may not contain faces at all. This is especially likely in real-life applications. This poses a problem for the systems that assume that a face is always present.

The second step is to localize all salient features that are needed to classify the expression. Most models first locate one of the salient features. The eyes are usually the first to be located because the high contrast observed at the outer rim of the iris makes them relatively easy to locate. An internal model (template) stores the location of all other landmark points relative to the eyes. This is how all other landmark points are found. This approach is vulnerable to the problems of poor illumination, varying angles of the observed face (posure) and partial occlusion of the face. Illumination can be a problem because we are essentially looking at pixel values. More specifically, we are trying to localize the facial features by using contrasting pixel values to determine certain landmark contours or points such as the silhouette of the face, corners of the mouth or tip of the nose. Poor illumination will diminish the contrast and consequently increase the difficulty of finding these landmarks. The second problem is that, in real life, the pose of a persons face will not necessarily be from a frontal angle. Because most models that are used to map the landmark points assume a frontal view, observing a face from a certain angle results in poor performance. As explained, a specific single landmark point is first located. The next step is to locate all other landmark points by determining their location relative to the first point, but this is where things go wrong because the relative locations are not the same as they would be from a frontal view. Consequently the landmark points are misplaced. Finally, most models assume full visibility of all facial features (eyes, mouth etc). However, when a part of the face is obscured for instance by a hand, facial hair or cap, we have the same problem: the observed face does not fit the assumed model and landmark points are incorrectly placed.

Finally we have to deduce the correct emotional state from the observed facial features. The analysis of the facial features is often achieved by a probabilistic model because classification without error is assumed to be impossible. A human face can exhibit complex and intricate expressions. Facial expressions are dependent on many factors such as muscle contractions, current emotional state and its implied context. Furthermore, facial expression are individually independent: no two people exhibit the same expression in the same way. These factors make the recognition of facial expressions a challenging task. One possible approach is to link the positions of the observed key points directly to a particular expression. For example, an opened mouth and raised eyebrows could correspond to an expression of surprise. Another approach labels certain areas of the face separately and assign expressions to certain combinations of observed labels.

One of the other prominent topics in facial expression recognition is the availability of useable data to train the introduced models. Bayesian networks gain their classification abilities by learning from examples. This means that we must gather examples that capture the entire span of possible expressions/ faces. This is a time consuming task and requires a significant amount of expertise. For this reason only very little data is publicly available and most models are trained on their own small data set. This makes comparison between models particularly difficult. In a nutshell: lack of data results in poor performance of expression recognition models, and for Bayesian networks in particular.

1.4 Survey focus and objective

In this survey we will concentrate only on the process of facial expression analysis as we explore ways of using probabilistic models for determining observed emotional states. We will discuss literature that introduces systems that emotional labels for observed faces. This area of research is currently very active, as *context sensitive reasoning* systems are increasingly popular. Human-Computer Interaction often suffers from a lack of context. Facial expression recognition is a powerful way to determine context, and thus to enhance HCI. As example, consider the automated agent discussed in the previous section. It uses the observed facial expression to determine the context, providing different kinds of assistance depending on the observed emotional state of the user. Occasionally we will pay a small amount of attention to face detection and extraction of facial key points for reasons of clarification, but this is not where we lie our main focus.

We will discuss one such a probabilistic model in particular: the Bayesian network. Bayesian networks is an increasingly popular approach towards problems of uncertainty. It learns to estimate probability in a heuristic way by adjusting its parameters to a set of training examples. However, like each classifying method, it comes with it own set of merits and drawbacks. One particular drawback can be found in the small sample case, which means that only a very limited amount of training samples are available. Unfortunately this seems to occur on quite a regular basis, as data gathering and classification are a time-consuming tasks.

Consequently several approaches of sample based learning in the small sample case have been proposed, each using different techniques and insights. The objective of this survey is to answer the question: *Is there still a future for Bayesian networks when it comes to facial expression techniques?* Obviously, Bayes is not the only method available and other techniques are perhaps superior for this particular kind of application. Yet there are several ways in which Bayesian networks can be applied, depending on how the input is pre-processed, which structure is used, which assumptions are made concerning dependencies etc. Each separate approach may have its own merits and drawbacks and may yield different results. One of the strengths of Bayesian networks worth mentioning is that it can learn even with incomplete data, meaning it is able to 'fill' the gaps in examples that are used for training. This is the direct consequence of the assumed dependence between separate variables

This is a literature survey on the application of Bayesian networks to facial expression recognition. Chapter two will give a brief overview of Bayesian networks and how they can be applied to model uncertainty in real-life problems. Chapter three will discuss some articles that describe different attempts to apply Bayesian networks to facial expression recognition. Chapter four will handle some articles that are relevant to the small sample case. Some criticism, conclusions and final thoughts are given in chapter five.

2 Bayesian Networks

This chapter will handle the theory of Bayesian networks in general [3, 4, 5]. Although a number of algorithms and techniques are mentioned, it is beyond the scope of this survey to discuss each of them in more detail.

2.1 Bayes rule

When we try to model uncertainty perceived in our world we strive towards a model that is simple enough to remain computationally tractable, yet complex enough to remain useful. The source of complexity in computing uncertainty is dependency between different events. In order to understand the relation between different events, we need a way to calculate these dependencies.

Science has long favored probability as a model for uncertainty. In 1819, Pierre LaPlace said "Probability theory is nothing but common sense reduced to calculation. This chapter will discuss one type of probability model: the Bayesian Belief Network. Thomas Bayes was an 18th-century cleric who first used probability inductively and established a mathematical base for probability inference. He set down his findings on probability in "Essay Towards Solving a Problem in the Doctrine of Chances" (1763) published posthumously in the *Philosophical Transactions of the Royal Society of London*. Reverend Thomas Bayes contributions were immortalized by naming a fundamental proposition in probability after him, called Bayes' Rule:

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Here follows a short explanation of Bayes' Rule: equation [2.1] states the if we want to know the probability that hypothesis *H* is true given evidence *E*, we need to calculate the probability of *E* being true given that *H* is true, multiplied by the probability that *H* is true, normalized by the probability of *E* being true. In short: of all the times of *E* being true, how often is *E* caused by H?

For instance, imagine we have a black box of which the contents are invisible to us, but we know that there only 3 possible configurations for the contents of the box: it either contains [configuration 1: 3 white marbles and 1 black marble], [configuration 2: 2 white marbles and 2 black marbles] or [configuration 3: 1 white marble and 3 black marbles]. Each of the three configurations is equally likely to be true, so P(configuration 1) = P(configuration 2) = P(configuration 3) = 1/3. This is known as the *a priori* probability, meaning the probability of a hypothesis being true before any evidence is apparent. In Bayes' rule the *a priori* probability is represented by P(H). It is given that the world contains just us much white marbles as it does black marbles. Now we take a single marble out of the box and it turns out to black. Intuition tells us that it is no longer equally likely for each of the configurations to be true, since it is obviously more likely to draw a black marble from configuration 3 than it is to draw a black marble from configuration 1. Bayes' rule now allows us to calculate the *a posteriori* probability, meaning the probability of an hypothesis after all evidence is taken into account. In Bayes' rule the *a posteriori* probability is represented by P(H|E). Applying Bayes' rule we can calculate that P(configuration 1) is now 1/2, P(configuration 2) = 1/3 and P(configuration 3) = 1/6.

Although this rule might seem fairly trivial at first, it is the cornerstone of probability inference. Bayes Rule is so important because in the physical world we often have no specified knowledge about P(H|E) but we do have some knowledge about P(E|H), P(H) and P(E). For instance, we might want to guess the probability that a person has the flu, *given that he has a cold*. Assuming we don't have any particular information for this phenomenon, all we need to do in order to make an educated guess is to specify the probability that a person has a cold, given that that person has the flu, the independent probability that a person has the flu and the independent probability that a person has a cold, which can usually be derived from medical records.

2.2 Knowledge in an uncertain domain

We use random values to model uncertainty in the world around us. For instance if we toss a coin we might consider the side on which the coin will fall as a random value between 0.0 and 1.0, a value between 0.0 and 0.5 indicating heads and a value between 0.5 and 1.0 indicating tails. Here we have only the single variable coin, but we can easily imagine a problem domain

with several different variables. The complete set of random variables used to describe this problem domain is called the full joint probability distribution. The full joint distribution specifies the probability of every atomic event (every possible combination of variables) and is therefore a complete specification of the uncertainty within the problem domain.

To explain, consider a day in Paul's life. Since Paul is particularly fond of his hair, we want to say something about whether he is having a good or a bad hair day, and whether his hair actually looks good or not. We also take into account if Paul still has enough gel to perform maintenance. Coincidentally, he also has a an important job interview and a date with a particularly nice girl on the same day, and we want to say something about the success of both. Now let us assume we have Boolean variables, *hairLooksGood, runOutOfGel, badHairday, dateSuccess* and *interviewSuccess.* We know that if we are to answer any question within our problem domain, we have need of the full joint probability distribution. This is where things go awry. First of all, this approach becomes computationally intractable as the number of variables grow, since each variable has a dependency with each other variable. The full joint distribution in the example above would have 2^5 entries, but one can imagine the increasing complexity if we should add another variable or if we use more than 2 possible values for each variable. Secondly, specifying data for all atomic events is rather difficult as it requires vast amounts of data which are almost never available. So we need another way to model our problem.

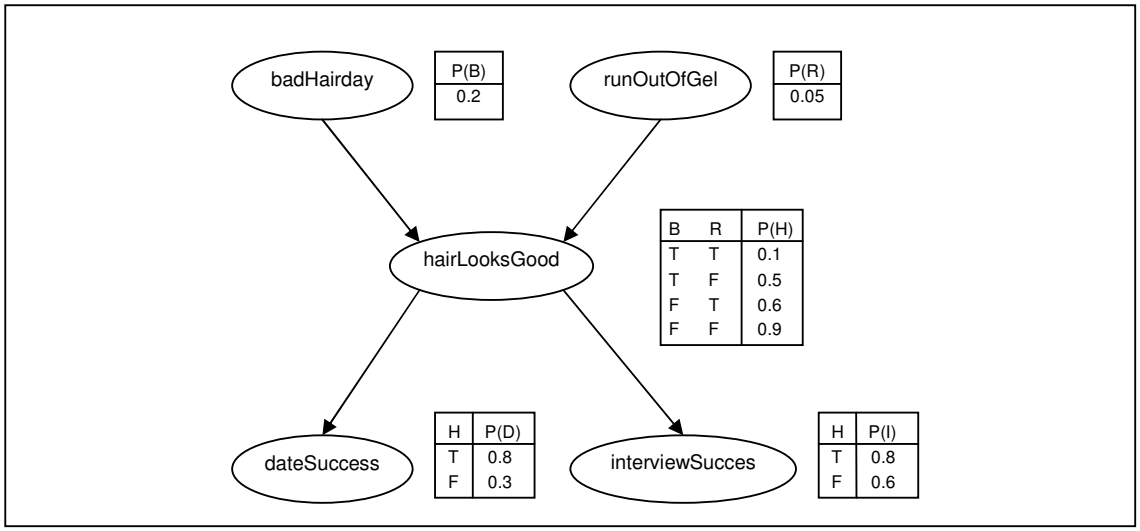

Figure 2.1: A typical Bayesian network

2.2.1 Network topology

Here we introduce the Bayesian network, as shown in figure 2.1. In a Bayesian network only a few dependencies are modeled, since we only model the dependencies between variables that directly influence each other. The rest of the variables are assumed conditionally independent. This greatly reduces the complexity of our model yet it can still give a concise specification of any full joint probability distribution [3]. A Bayesian network is a directed graph in which each node is annotated with quantitative probability information. A full specification is as follows:

1. A set of random variables makes up the nodes of the network. Variables may be discrete or continuous.
2. A set of directed links connects pairs of nodes. If there is an arrow from node X to Y, X is a parent of Y.
3. Each node has a conditional probability distribution that quantifies the effect of the parents on the node.
4. The graph has no directed cycles and hence is a Directed Acyclic Graph (DAG).

The topology of the network specifies the conditional independence relationships that hold in the domain. The intuitive meaning of an arrow from X to Y is that X has a direct influence on Y. For instance, figure 2.1 shows that *hairLooksGood* has a direct influence on *dateSuccess*. On the other hand we have modeled that the success of Paul's date has no effect on whether or not his job interview is a success. While it is true that both variables have a strong dependency on the state of Paul's hair, they do not have any effect on each other. In the full joint probability distribution there is nothing much we can do with this intuition since

the probability for each atomic event must be specified, but in a Bayesian network we can make the two variables conditionally independent simply by not connecting them directly.

Let us take another look at the topology of the network. We see that whether or not Paul still has sufficient gel and whether or not he is having a bad hair-day directly affects the looks of his hair. In turn, the looks of his hair have a direct effect on the success of his date and his job interview. Thus the network assumes that the success of Paul's date depends solely on the looks of his hair on is conditionally independent of *badHairday* and *runOutOfGel* in a direct sense. Furthermore, the network assumes that success of Paul's job interview has no effect on the success of his date later that day.

Notice that the network ignores a lot of other factors that are relevant in real life. For instance, Paul's resume and communication skills will have a certain effect on the success of his job interview, but instead of making these factors explicit in the network, we assume they are implicit in the probability distribution of the node *interviewSuccess*. Therefore we can interpret the probability in a node as the summarized probability of all possible relevant factors (i.e. resume, communication, aptitude for the vacancy, mood of the interviewer etc.) which are not explicit in the network. The reasons for these factors to remain implicit is that usually it's just to difficult to obtain the relevant data to model these variables explicitly, assuming it's even possible.

2.2.2 Conditional probability tables

Now let us turn to the conditional distributions shown in figure 2.1. In the figure each distribution is shown as a conditional probability table (CPT). This form of table is used for discrete variables. Each row in a CPT contains the conditional probability of each node value for a conditioning case. A conditioning case is simply a combination of values for the parent nodes and can be seen as a miniature atomic event. The sum of each row must be 1, because the entries represent an exhaustive set of cases for the variable. For Boolean variables we know that if the probability of a true variable is *p*, the probability of a false value is *1-p*. For reasons of efficiency we have omitted the second value in the CPTs.

A Bayesian network provides a complete description of the domain, meaning that every entry in the full joint probability distribution can be calculated from the information in the network. A generic entry in the fully joint probability distribution is the probability of a conjunction of particular assignments to each variable, such as $P(X_1=x_1 \wedge \ldots \wedge X_n=x_n)$, or $P(x_1, \ldots, x_n)$ in abbreviated form. The value of this entry is given by:

$$P(x_1,...,x_n) = \prod_{i=1}^{n} P(x_i \mid parents(X_i)),$$

Where *parents*(X_i) denotes the specific values of the parent nodes of X_i. Put into words, each entry in the full joint distribution is represented by the product of the appropriate probability entries in the conditional probability tables in a Bayesian network. To illustrate, say that we want to know the probability that Paul has great looking hair(h), while he has not run out of gel (¬r) and does not have a bad hair-day (¬b). Furthermore his job interview fails(¬i), but his date is a great success(d):

$P(h \wedge \neg r \wedge \neg b \wedge \neg i \wedge d)$
$= P(\neg i|h)\, P(d|h)\, P(h|\neg r \wedge \neg b)\, P(\neg r)\, P(\neg b)$
$= 0.2 \times 0.8 \times 0.9 \times 0.95 \times 0.8 = 0.10944$

It can be shown that a correctly constructed Bayesian network represents the full joint probability distribution [3], which means that it can be used to answer any query within it's domain, by summing the probabilities of all the atomic events within the specifications of the query. Later in paragraph 2.4 we will discuss how to derive the probability of an arbitrary variable, given the values of (some of) the other values.

2.2.3 Network construction

Now that we have some understanding of how a Bayesian network works and what implications are made by the topology of a network, we will pay some more attention to the construction of a Bayesian network. As well as being a complete and non-redundant representation of the domain it is usually also a far more compact representation of the problem domain. This compactness is due to the fact that a Bayesian network is a locally structured system. This means that each subcomponent interacts directly with only a limited number of other subcomponents, regardless of the total number of subcomponents. In a Bayesian network it is reasonable to assume that a subcomponent (node) is influenced by at most *k* other subcomponents (parent nodes). In the case of Boolean values that means that each CPT will contain at most 2^k entries, so if we have *n* subcomponents (nodes) that brings us at a total of $n2^k$ entries. In contrast, a full joint probability distribution needs 2^n entries.

There are domains in which case each variable can be influenced by each other variable. In this case we need just as much information as we need to specify the full joint probability distribution. However, in general, the increase in accuracy of such a system does not weigh up to the increase in complexity and need for more detailed information gathering. Therefore we often reduce complexity by assuming conditional independence between variables that have only a marginal dependency.

Since we know that the topology of a network has some serious effect on how the dependencies between variables are modeled, we can also see that it is not a trivial matter to properly construct a Bayesian network. We do not only need to limit the number of dependencies (i.e. limit the number of parents for a particular node) but we have to choose the topology in such a way that variables become the parents of the variables that they directly influence, without becoming a cyclic graph. Therefore the 'correct' order of constructing a network is to first add the nodes that contain the *root* causes, then the variables they influence directly and so on, until we reach the leaf nodes which contain variables that have no direct causal influence on any of the other variables. If we should choose the wrong order in building a network this does not have any influence on the correctness of the model, but it does usually require more conditional probability tables entries in total, and, what's worse, in order to fill the conditional probability tables we often need tenuous information which requires difficult and unnatural probability judgments.

In general, we should try to find the network structure that best fits the joint distribution of all the variables given the available data. Another essential issue is that, given that every dataset is finite, we should be careful not to overfit the network to the data, which means that the generalizing ability of the network has completely disappeared. Several automated methods are available for this problem, such as the Stochastic Structure Search Algorithm proposed in [18]. We conclude this paragraph with the notion that any topology is correct in that it describes the exact same full joint probability distribution. However, if we stick to causal relationships during construction we end up with compacter models of which the conditional probability tables are simpler to specify.

2.2.4 Naïve Bayes

One variation of the Bayesian network that is used often is the Naïve Bayesian network. This type of network basically assumes that the observed input variables are dependent only the response variable. The remarkable thing about Naïve Bayes is that it assumes conditional independency between its predictive variables given the state of its input variables. Nevertheless, Naïve Bayes performs remarkably well in practice, which fuels the argument that increased complexity is not always a necessity for good performance. Furthermore, the simplicity of the Naïve Bayesian network is appealing because we don't have to make estimations for countless parameters. Using Naïve Bayes is a trade-off: the incorrectness (assumed independency of observations) of the underlying model is countered by the limited error in parameter estimation (due to the limited number of parameters).

For example, the Naïve Bayesian network in figure 2.2 implies that if we observe that our girlfriend has wild mood swings and has a suspicious sudden craving for pickles, this will tell us nothing about whether or not she will suffer from morning sickness, although unfortunately we know better in practice.

Performing inference in a Naïve Bayes network differs slightly from inference in a normal CPT network as the cause and consequences are reversed. Inference can be achieved as follows:

$$P(pregnant \mid m,c,w) = \frac{P(m \mid pregnant)P(c \mid pregnant)P(w \mid pregnant)P(pregnant)}{P(c)P(m)P(w)},$$

which can be generalized to:

$$P(c \mid x_1,...,x_n) = \frac{\prod_i P(x_i \mid c) \cdot P(c)}{\prod_i P(x_i)}$$

Naïve Bayes is especially applicable in situations when there is not enough data to detect the conditional relations between variables. In such a case Naïve Bayes can still achieve good classification results. Besides Naïve Bayes, other similar network structures are available such as Tree-Augmented Naïve Bayes described in [16, 17, 21] which is similar to Naïve Bayes, but allows each observed feature to have a single extra parent feature beside the parent node of the response variable. In this way the features are not totally independent, but the complexity of the network does not increase significantly.

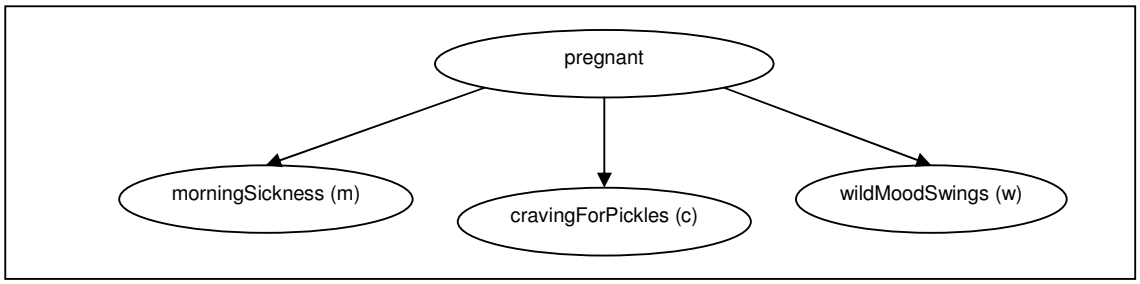

Figure 2.2: A Naïve Bayesian network

2.2.5 Continuous variables

Up to this point we have only used Boolean variables in our examples, but many real world problems involve continuous variables that cannot be specified discretely, such as temperature, distance or the configuration of facial features. As continuous variables have an infinite number of possible states, we can see that building a conditional probability table would be a bad idea. There are two ways to tackle this problem. The first is by using discretization, dividing the possible values into a fixed set of intervals. The second is by defining a probability density function that is specified by a finite number of parameters. Gaussian and normal distributions are often used for this purpose, which only require the mean (μ) and variance (σ^2) as parameters.

A network that incorporates both discrete and continuous variables is called a hybrid Bayesian network. Such a networks is proposed in [19]. In the case of hybrid networks we need to specify new distributions for two cases: continuous variables that have discrete and/or continuous parents and discrete nodes that have continuous parents. We do so by the following example: it is Friday night and Paul feels like going out to dinner in his favorite restaurant. The restaurant employs two different chefs, one of which works considerably faster than the other. Let us assume that the time between ordering his favorite plate and his food being served depends on which of the chefs is working that particular night and the number of customers that is currently in the restaurant. Whether or not Paul will wait depends on the estimated waiting time. The described problem is modeled in figure 2.3.

The *WaitingTime* variable has both a continuous and a discrete parent. To handle *NumberOfCustomers* (*n*) we specify how the distribution over the waiting time *w* depends on the continuous value *n*.). More explicitly we define a function *f(n) = average w*.

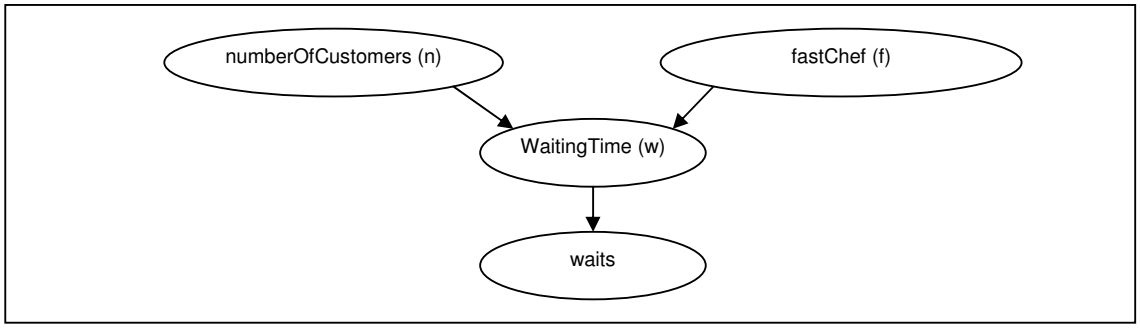

Figure 2.3: Network with discrete and continuous variables

To handle *fastChef* we specify a separate distribution for each possible value. So in this case we will specify both *P*(*waitingTime* | *numberOfCustomers*, *fastChef*) and *P*(*waitingTime* | *numberOfCustomers*, ¬*fastChef*). Using a linear Gaussian distribution we have:

$$P(w \mid n, f) = N(a_t n + b_t, \sigma_t^2)(w) = \frac{1}{\sigma_t \sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{w-(a_t n+b_t)}{\sigma_t}\right)}$$

$$P(w \mid n, \neg f) = N(a_f n + b_f, \sigma_f^2)(w) = \frac{1}{\sigma_f \sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{w-(a_f n+b_f)}{\sigma_f}\right)}$$

So for this example the conditional distribution for *waitingTime* is specified by the parameters by using the linear Gaussian distribution and providing the parameters a_t, b_t, σ_t, a_f, b_f and σ_f. Furthermore, when the prior probability of *fastChef* is known, we are able to average over the two possible distributions in order to specify a distribution for *P*(*w*| *n*).

When dealing with discrete variables that have continuous parents we use threshold functions. Consider for example the *waits* node in figure 2.3. We assume that Paul will wait if the waiting time is low and that he will depart if the waiting time is high and the that the probability of Paul waiting varies smoothly somewhere in the middle. Two of the possible threshold functions are the probit and the logit distribution. The first uses the integral of the standard normal distribution, the latter uses a sigmoid function as shown below:

$$P(waits \mid waitingTime = w) = \frac{1}{1 + \exp(-2\frac{-w+\mu}{\sigma})}$$

Although the probit distribution is usually a better fit to real situations, the logit distribution is computationally less demanding and therefore often preferred.

2.3 Inference in Bayesian networks

The purpose of a probabilistic model is that we can determine the posterior probability distribution for a set of query variables given a set of observed events. We achieve the posterior probability distribution by using inference within the network. Exact inference can be intractable for computation so in practice approximate inference is used. An explanation as to why exact inference proves to be intractable can be found in [3]. We will take a short look at both exact and approximate inference.

2.3.1 Exact inference

Exact inference can be achieved by enumeration. Given the problem as described in figure 2.1, we might want to know P(*badHairDay*| *dateSucces=true*, *interviewSucces=true*). To get our answer we enumerate over all possible values of the variables that are *not* observed (*hairLooksGood*, *runOutOfGel*). In order to get the probability distribution for our query variable we need to separately enumerate the probabilities of all atomic events that *badHairDay* is true and that *badHairDay* is false:

$$P(b \mid d,i) = \alpha \sum_r \sum_h P(b)P(r)P(h \mid b,r)P(d \mid h)P(i \mid h) = \alpha \cdot 0.08016,$$

and

$$P(\neg b \mid d,i) = \alpha \sum_r \sum_h P(\neg b)P(r)P(h \mid \neg b,r)P(d \mid h)P(i \mid h) = \alpha \cdot 0.46968,$$

so we have found that:

$$P(B \mid d,i) = \alpha < 0.08016, 0.46968 > \approx < 0.15, 0.85 >$$

The parameter α acts as a normalization factor. The enumeration method specified by this example is correct, but terribly inefficient as the same calculations are computed several times. Other, more efficient methods for exact inference are available, which store the intermediate results to save computation. One such an algorithm is the *variable elimination algorithm* described in [3]. We will not explicitly specify the algorithm here, but suffice it to say that it operates by the use of variable elimination, which means that the algorithm evaluates an expression right-to-left, storing intermediate results and summing over each variable only with the portion of the expression that depend on that variable.

2.3.2 Approximate inference

Given the intractability of exact inference it is important that we consider approximate inference methods. To do this we can use random sampling algorithms also known as Monte Carlo algorithms. Monte Carlo methods are widely used in computer science to estimate quantities that are difficult to calculate exactly. Several Monte Carlo methods are available to achieve approximate inference in Bayesian networks: *direct sampling*, *rejection sampling*, *likelihood weighting* and the use of *Monte Carlo Markov Chains*. In this paragraph we will only discuss direct sampling, but detailed descriptions of each method can be found in [3].

Perhaps the most simple Monte Carlo method for approximate inference is direct sampling. We illustrate how this works with the aid of figure 2.4 assuming the ordering [sunnyDay, eatIcecream, drinkSoda, tummyAche]:

1. Sample from *P*(*sunnyDay*) = <0.5, 0.5>; suppose true
2. Sample from *P*(*eatIcecream*| *sunnyDay=true*) = <0.7, 0.3>; suppose true
3. Sample from *P*(*drinkSoda*| *sunnyDay=true*) = <0.9, 0.1>; suppose false
4. Sample from *P*(*tummyAche*| *eatIcecream=false*, *drinkSoda=true*) = <0.15, 0.85>; suppose false

Now we have a sample [true, true, false, false]. Suppose we want to know the probability *P*(*true, true, false, false*) using direct sampling. We first take *N* samples as described above. Then we check to see how many of those samples match [*true, true, false, false*]. The probability is then calculated by dividing the number of matching samples by the total number of samples. It can be shown that if *N* is large enough, the calculated probability converges to the expected probability [3]. So if we take 1000 samples we expect about 30 (0.2975 %) samples to match the sample [*true, true, false, false*] which gives us *P*(*true, true, false, false*) = 0.003.

Rejection sampling in Bayesian networks uses the same technique, but it rejects all samples that do not match the evidence. So if we want to know *P*(*X=x*|*e*), all samples that do not match *e* are discarded so that in the end we have the frequency of *X=x*

within all the samples that *e* is observed. Suppose that we take 1000 samples and we discard 533 as they do not match *e*. In the remaining 467 samples we observe *X=x* 120 times. So in this way we obtain $P(X=x|e) = 120/467 \approx 0.26$.

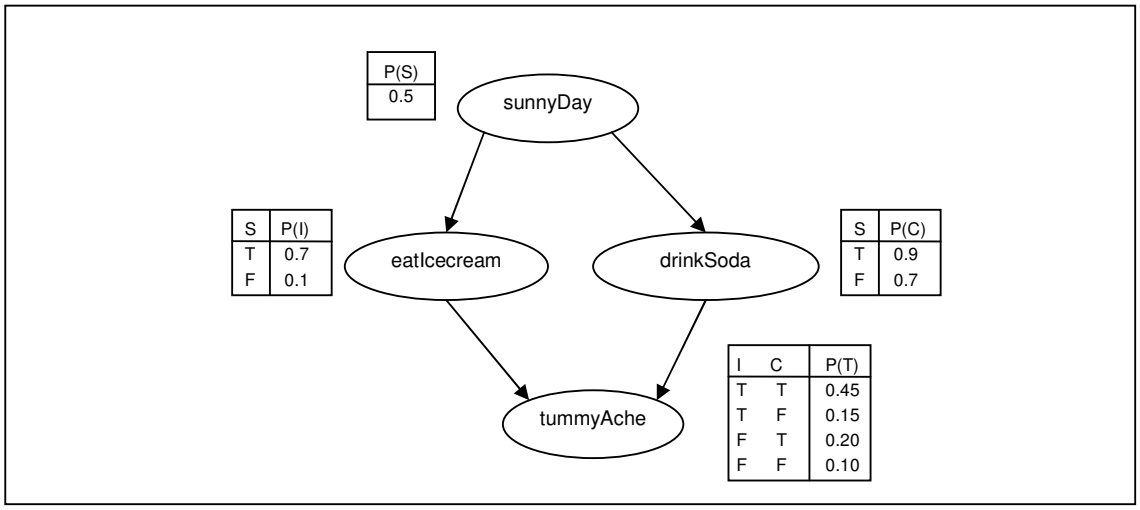

Figure 2.4: Bayesian network

Likelihood weighting differs from rejection sampling in that it generates only samples which match the evidence *e*, hence there is no need for rejection and consequently making it more efficient. Monte Carlo Markov Chain method differs from all the other methods in that it does not generate samples from scratch, but generates a new sample by adjusting the previously generated sample by changing one of the unobserved variables.

2.4 Learning in Bayesian networks

In order for a Bayesian network to give an accurate conditional probability distribution we need to specify all prior and conditional probabilities i.e. fill all conditional probability tables. Several methods are available for finding the parameters for a network but not all are equally suited. Statistical learning often uses methods such as Maximum likelihood parameter learning, which chooses the parameters which best fit the available data set. The catch is that when you have little or incomplete date you often get misleading parameters.

2.4.1 The beta distribution

The Bayesian approach to parameter learning places a hypothesis prior over the possible values of the parameters and updates this distribution as data arrive. Suppose we like a particular kind of chewing gum. Sometimes when opening the gum packaging we find a collectible trading card, but whether or not the gum comes with a card cannot be observed before unpacking. Now suppose we want to specify the probability that we find a collectible trading card, that is $P(card) = \theta$. In the Bayesian view θ is the (unknown) value of a random variable Θ. Thus $P(\Theta = \theta)$ is the probability that we have a chance of θ to find a trading card.

To specify θ we use the beta distribution. Each beta distribution is specified by the two hyperparameters *a* and *b*. They are called hyperparameters because they are in fact parameters of a distribution which in turn is also parameter. So we have:

$$beta[a,b](\theta) = \alpha\theta^{a-1}(1-\theta)^{b-1} \text{ in which } \alpha = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} = \frac{(a+b-1)!}{(a-1)!(b-1)!},$$

for θ in the range [0, 1]. The parameter α acts as a normalization factor depending on *a* and *b*. The mean value of the distribution is $a/(a+b)$. To show how this works suppose *a* counts the number of occurrences that we do find a collectible card in a package and *b* counts the number of times that we don't. Given the mean value $a/(a+b)$ a higher value of *a* suggest a value

closer to 1. Larger values of (a+b) make the distribution more peaked, suggesting a greater certainty of the value of Θ. Furthermore if Θ has a prior beta[a, b], then, after a data point is observed, the posterior value of Θ is also a beta distribution.

For instance, suppose we have Θ with prior beta[a, b] and observe a gum with an included card then the probability $P(\theta|\ card)$ = beta[a+1, b](θ). Thus after observing a gum package with a card, we simply increment hyperparameter a in order to get the posterior. In the same manner we would have incremented hyperparameter b in the event that we had *not* found a card with the gum. So we can see the hyperparameters as virtual counters for each of the possible events. Furthermore, as the number of counts increases, the beta distribution converges to a narrow peak such as shown in figure 2.5, where the true probability of finding a trading card is equal to 20%.

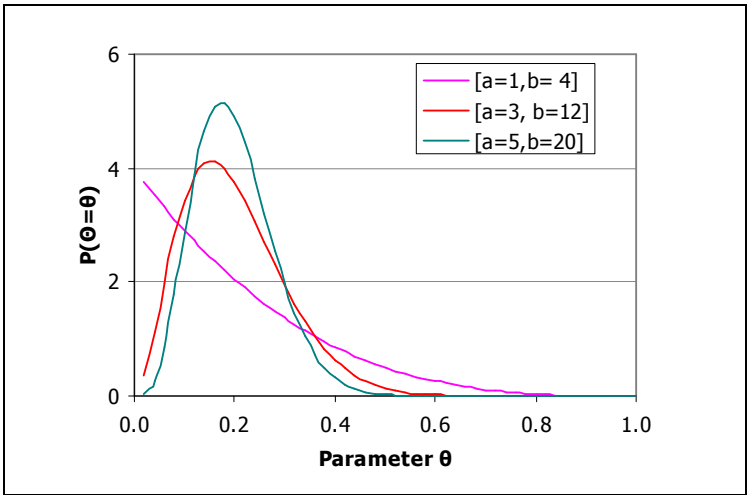

Figure 2.5: Beta distributions

The beta distribution takes only two possible values of a variable, such as is the case with Boolean variables, but other distributions such as the Dirichlet family are distributions of discrete multivalued parameters. The family of Normal-Wishart distributions can be used for the parameters of a Gaussian distribution.

2.4.2 Hidden variables

Up until now we have only discussed fully observable case, but in real world problems we are not always able to observe every relevant variable. Of course we may ask ourselves, if we don't observe these variables, do we really need them? When we compare figure 2.6a and 2.6b we can see that hidden (latent) variables can drastically reduce the complexity of a network. Each of the variables has 3 possible values. The network in 2.5a results in a total of 76 independent parameters, while the network in figure 2.5b requires only 34 independent parameters.

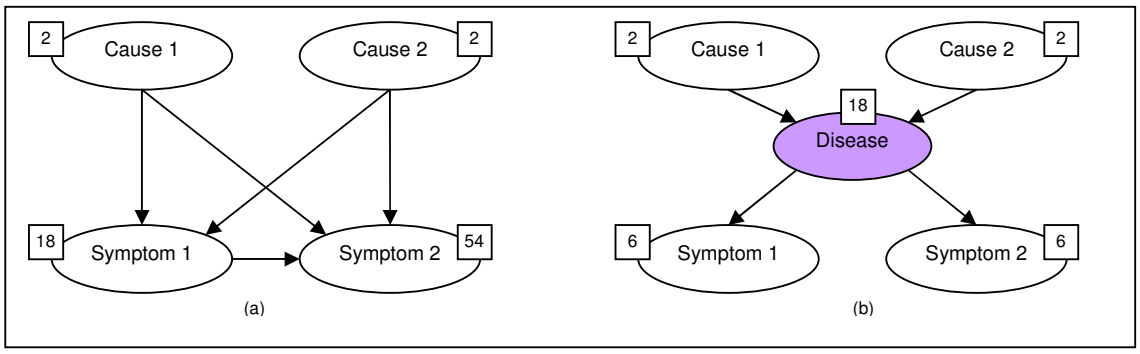

Figure 2.6: Bayesian network with a hidden variable (b) and the same network without (a). Each node is labeled with the number of independent entries in the conditional probability table of that node.

So using hidden variables means that the amount of data needed in order to specify a Bayesian network is reduced. Evidently hidden variables are important, but they do make learning parameters more complicated as it is quite difficult to determine the conditional probability table for a variable of which the value is not represented in the training set. In order to estimate the conditional probability distribution of a hidden variables the Expectation-Maximization algorithm is often used. When applied to Bayesian networks, this algorithm operates in two steps: first it *pretends* to know the conditional probability distribution and then we use inference to update the conditional probability distribution until it converges to the true conditional probability distribution. For the general case in which we are learning the conditional probability tables for each variable X_i, given its parents, the update is given by the normalized expected counts as follows:

$$\theta_{ijk} \leftarrow \hat{N}(X_i = x_{ij}, Pa_i = pa_{ik}) / \hat{N}(Pa_i = pa_{ik}),$$

where i is an index for the variable, j is an index for its possible values and k is an index for the parents of X_i. The general idea is that a Bayesian network uses only a single algorithm for both learning parameters of (both hidden and observed) variables and executing queries: inference.

3 Facial Expression Recognition

The aim of facial expression recognition is to analyze the affective state of the observed subject. Generally, the face is analyzed by identifying so-called facial features (eyes corners, mouth corners etc.). All or a combination of these features are then used as input for a classifier e.g. a Bayesian network which attempts to assigned some emotional state to the observed features. This chapter discusses the Bayesian network applied to facial expression recognition.

3.1 History of facial expression recognition

Images of faces have long been an subject of fascination in the scientific study of emotion. From relatively general studies in the first half of the 20th century to contemporary research in relation to neuroscience, the human face seems to remain the focal point of many endeavors to classify facial expressions. The publication [6] describes the development of facial expression classification starting with early research in the 1920s and 30s.

One such a study was the study by Antonia Feleky which had the objective to show that a relation existed between certain emotional states and certain facial expressions. Volunteers were required to look at 86 photos of a woman portraying specific facial expressions and were asked to note the expression (a list of 100 possible expressions was provided) that was suggested to them for each single photograph. The 'Feleky' study served as a baseline for later investigations of facial expression and emotion, such as the research by C.A. Ruckmick who claimed that the interpretation of the perceived emotion depended not only on the photograph but also on the emotional state of the beholder. Other research by L. Kanner points out the linguistic hazards in interpreting emotions from facial expressions, as the observer may not have the vocabulary to express exactly what he or she sees on a photograph.

Although early research focused on the *context of origin* (emotional cause), later research seemed much more concerned with the *context of application* such as is the case in expression recognition. It was not until the 1970s that a system for measuring and analyzing facial expression was developed, when knowledge of the psychological affect of facial expressions and the knowledge of the anatomy of the face were combined. Psychologists Paul Ekman, Richard Sorensen and Wiliam Friesen had just completed a set of cross-cultural studies which suggested six basic emotional prototypes, as a handful of images consistently received high correct classification rates, independent of literacy, media exposure and origin of the observer. These emotional prototypes were: joy, sadness, fear, disgust, anger and surprise. In 1978 Ekman and Friesen [7] published a coding scheme based on these images in a monograph titled *The Facial Action Coding System* (FACS).

3.2 Facial Action Coding System

FACS is meant to provide a standard baseline against which researchers could test their hypotheses regarding the relation between expression and emotion. It aims to describe all visually discernable facial movement on the basis of 44 unique Action Units (AU). Among the 44 AUs, 12 describe the contractions of the muscles in the upper part of the face and 18 in the lower part. The other 14 are miscellaneous actions that do not have a specific anatomic basis. The general idea is that tracking stationary points on the face of a subject is not very useful, since the facial features vary wildly over race, age and sex. Instead FACS relies on the dynamics (hence the name 'action' unit) between these stationary points (contractions of muscle groups) which is presumed to be universal. In fact, FACS is probably the best known study on facial expressions and is still widely used by researchers in the area of facial expression recognition today, as numerous studies use FACS either to code already existing images or use it to create new expressions on a computer.

When data is coded by FACS (labeled data), or an alternative coding scheme, various research studies have achieved better recognition results then when using un-coded or unlabeled data. The reason might be that classifiers can be better trained when specific facial dynamics are linked with specific classes, whereas raw data i.e. a set of pixel values is harder to classify. Paper [21] shows that better results are achieved when training a Bayesian network exclusively with labeled data.

Although numerous studies gratefully use the coding of FACS, not much research has been done to the reliability of the coding of facial expressions when applied to spontaneous expressions. One point of criticism that is often encountered is that

spontaneous expressions differ from voluntary expressions, which mean that classifiers might perform poorly if trained on voluntary expressions and applied to spontaneous expressions. So we might ask if FACS is capable enough to accurately code spontaneous expressions. In order ensure this, we need to be sure that all FACS coders would code a stream of spontaneous facial images in the same way.

An evaluation of FACS [8] expresses concern about four different types of reliability between different observers. The first is the reliability of the correct AU being identified. Most studies claim to have a good average reliability but little is know of reliability for specific AUs. The second reliability expresses concern about the temporal aspect of facial expressions in video streams. Transition from one expression to another goes smoothly, so we might ask if all FACS coders choose the same frame (moment) at which the AU has become discernable or has disappeared. The third type of reliability concerns the degree of intensity of the observed AU. The FACS includes five levels of intensity for each AU, but it may be expected that different coders have different thresholds for transition between a level of intensity. The fourth type of reliability concerns different FACS coders attributing the same AUs to specific prototype expressions.

To make an evaluation of the ability of FACS to model spontaneous expressions three different setups were used, all having multiple coders having to code the same video sequences. The results in paper [8] show that there is a high inter-observer reliability for which (spontaneous) AU and corresponding intensity is observed. The variance in timeframes of AU detection was in roughly 1/6 second, which is adequate for most expressions, but can be troublesome for some specific facial dynamics such as the blink of an eye. Also the attribution of AUs to specific emotional states was similar for all coders. In short, the overall results are pretty good, but not conclusive, as they are based on only three small experiments.

Other ways to code facial expressions include the Facial Action Scoring Technique (FAST: Ekman, Friessen & Tomkins 1971), A System for Identifying Affect Expressions by Hollistic Judgment (AFFEX: Izard, Dougherty & Hembree 1983) and the Maximally Discriminative Facial Movement Coding System (MAX: Izard 1979).

3.3 Cohn-Kanade database

Within the past decade significant effort has been put into methods for facial expression analysis. Because most research uses its own limited training data to measure performance, little is known about how different methods compare to one another. Furthermore limited data leads to specialized models that may perform well in a controlled environment, but will perform poorly when applied to a real-life environment. The article [9] describes the problem space for facial expression analysis. Not only the variance of the faces themselves are a problem but also the consistency of the observers analyzing and coding the data (e.g. with the Facial Action Coding System). Other problems are the difference between spontaneous and deliberate expressions such as in [8], head orientation and conformity of image acquisition. To overcome this problem a large database for facial expressions is proposed. In the literature this database is commonly known as the Cohn-Kanade database.

Figure 3.1: Frames from video sequences in the Cohn-Kanade database

The Cohn-Kanade database, officially baptized the CMU-Pittsburgh AU-coded Face Expression Image Database, consists of 1917 video sequences of 182 male and female subjects of varying ethnic background (81% Euro-American, 13% Afro-American, 6% other ethnic groups) between the age of 18 and 50. Furthermore all sequences are coded by the Facial Action Coding System for either target frames or the entire sequences. Additionally, each sequence was recorded from both a frontal view and a latent 30 degree angle. Although the database is under constant construction (e.g. addition of additional views and

information on emotional states) the Cohn-Kanade database is often used as testing and training data in contemporary research concerning the analysis of facial expressions.

3.4 Previous research

The study of human facial expressions is a very challenging form of pattern recognition as it focuses on non-rigid deformations of the face which differ from person to person. Humans seem particularly capable at recognizing facial expressions, but to create a system that is capable of the same is not a trivial undertaking. The Bayesian network is just one classifier in facial expression recognition and other approaches have been researched in the last couple of decades.

For example, paper [10] describes a model that is analyses face region motion by spatio-temporal analysis and then uses a *rule* based system instead of a probabilistic model to analyze the observed expression. Consequently the system is less flexible and less robust against movements of the head. Although [10] only describes rules to identify the six 'universal' facial expressions, it would be rather difficult to make a set of rules for each possible expression. It is however one of the early attempts to apply facial expression recognition on video streams.

The survey [11] gives a comprehensive overview of the research done in facial expression recognition in the period '90 to '00. Three distinct ways could be identified when applied to the analysis of observed facial features: the template approach, the rule-based approach and the neural network approach. The template-based approach attempts to match a newly observed face to a number of internal model representations (templates) and classifies the newly observed face to the label of the best fitting template. The rule-based approach uses an expert system to classify the newly observed expressions, using the basis of the observed facial features. The neural approach tackles the problem by training a neural network. Input images are translated to vectors which are used as the input for the artificial neural network. All methods are briefly discussed, but as the survey concludes, there is no accountable way to compare the methods as their reported performances are all based on training sets, varying both in size and content. Consequently it is very hard to determine which approach actually yields the best performance..

The Bayesian approach to facial expression recognition is relatively new as it has only been applied in the last 5 years. It can be seen as a combination of the rule-based and the neural network approach, as it makes us of a network but instead of being 'black-box', which is a property of neural networks, specific meaning is assigned to each node and the dependencies between separate input variables need to be specified explicitly, which is a property of rule-based systems.

3.5 Facial expression recognition and Bayesian networks

Automatic recognition of facial expressions is often approached as a process based on the analysis of stationary points (face contour) in contrast to transient points (facial features). But this is only one half of the process since, after identifying and coding the observed features, we want to determine the facial expression that is suggested by that particular set of facial features. To accomplish this we can use Bayesian networks. The hypotheses are the six prototypical expressions (or seven, if 'neutral' is included) and the observed Action Units such as found in FACS, are the evidence. The papers we will discuss in this section use different topologies for the proposed Bayesian networks. We discern: Naïve Bayes and Tree-Augmented Naïve Bayes.

Naïve Bayes

In the paper [12] a Naïve Bayesian network is used for the classification of facial expressions. A new approach is suggested as a Cauchy distribution is proposed instead of the Gaussian distribution, which is a common way to model continuous variables in Bayesian networks. The system starts by extracting the observed facial features. Instead of FACS, a more simple wireframe model is used to model the observed features, consisting of only 12 facial motion measurements. These features are then translated to an input vector X consisting of elements $(x_1, \ldots, x_{12})$. As a Naïve Bayes network assumes conditional independency between its input variables, which are in this case the elements of the input vector, the eventual problem is to model the probability $P(x_i|E)$ for each element, where E stands for one of the six prototype expressions. Now instead of using a Gaussian distribution to model $P(x_i|E)$, a Cauchy distribution is used. The motivation thought is that Cauchy might be better able to model outliers in the data set.

Two separate experiments are described: one of them person dependent, in which the test data contains only images from persons that were also part of the training set, and the other person independent, in which the test data only contains images from persons which were not a part of the training set. In the case of the person dependent experiment the Cauchy approach (80.05%) performs slightly better then the Gaussian approach (79.36%). In the case of person independent testing the performance was considerably worse, with Cauchy (63.58%) against Gauss (58.94%). The bad performance is attributed to the relatively small training set, problems with face tracking and the fact that some of the expressions have very similar facial measurements (such as fear and surprise). To overcome bad performance, [12] suggest categorizing facial expressions in [neutral, negative, positive, surprise] instead of the six prototypes normally used, as it can be argued that a lot of applications of facial expression recognition do not need more detailed categorization. Papers [14, 15] use the same alternative categorization. By using this alternative categorization, performance is considerably improved. The paper concludes stating that even though performance is quite poor it has been shown that Cauchy performs slightly better than Gauss in every situation. However, the proposed system in itself is does not yet perform well enough for real-world application.

The article [13] compares Naïve Bayes to two other methods for classification: Support Vector Machines (SVM) and Relevance Vector Machines (RVM). Although Naïve Bayes is a proven method for certain classification problems it shows that Naïve Bayes performs only as well as the size of the trainingset allows. This can be explained as follows: during training a Bayesian network models the dependencies of each feature on the given class (Naïve Bayes assumes no dependencies between features). In other words, during training the dependency distributions are learned. However, if the data is sparse, it may not span the entire distribution and the learned distribution may differ greatly from the true distribution, which results in poor performance of the Bayesian classifier. This is a common problem for Bayesian networks, as data is usually sparsely available. In chapter 4 we will pay more attention to this problem.

Support Vector Machined and Relevance Vector Machines both perform considerably better then Naïve Bayes in the experiment presented in [13] at the price of being computationally more complex. Also it must be noted that the experiment involved only static images for training. Both SVM and RVM aim at finding the optimal discriminate hyperplane in linear space. When facial features are translated to vectors in linear space this provides powerful classification techniques when little data is available, as both techniques are somewhat robust to overfitting in the small sample case. RVM is a refinement of SVM as it weighs relevant facial features more heavily which means it is better able to discriminate the features that have the highest degree of variation. This results in a smaller number of discriminative functions and consequently in lower computational demands which allows it to perform real-time. Real-time performance is important because many applications of facial expression recognition often operate in real-time.

Tree-Augmented Naïve Bayes

The paper [14] proposes to use a Tree-Augmented Naïve Bayesian network (TAN) for classification of facial expressions in video streams. The Tree-Augmented Naïve Bayesian network is similar to the Naïve Bayesian network with the difference that it allows a single additional parent (feature) for each observed feature. The motivating thought is that assuming facial feature independency is an unfound assumption in facial expression recognition. A TAN allows some dependency between variables. To determine which features have the strongest dependency, the correlation for each pair of features for each class is calculated. The performance of the proposed TAN is proves significantly better than that of the Cauchy-Naïve Bayes classifier in [12].

The paper [15] makes another comparison between different classifiers. The Tree-Augmented Naïve Bayesian network proposed in [14] is set out against a Naïve Bayesian network, an Artificial Neural Network (ANN) and a Hidden Markov Model (HMM). The first three approaches attempt to classify by analyzing static images from the observed video sequence, while the Hidden Markov Model handles the sequence as a single coherent input. Again, the motivating thought is that the observed features are not independent of each other and modeling additional dependencies might increase performance.

The experiment was performed using the Cohn-Kanade database [9] to ensure a large enough training set in order to make valid comparison. The proposed HMM approach could not be tested, since it required image sequences which start with a

neutral expression, transform to the emotional peak of the portrayed expression and then return to neutral state, while the image sequences in the Cohn-Kanade database end at the emotional peak. Results showed that the neural network achieved the best overall performance (73.81%), closely followed by the proposed TAN approach (73.22%). The Naïve Bayesian network was the worst performer (68.14%).

3.6 Bayes and Principal Component Analysis

A technique that is often used in conjunction with Bayesian networks in facial expression recognition is Principal Component Analysis (PCA). Principal Component Analysis, also known as the Karhunen-Loeve expansion, can be used to reduce the dimensionality of an input vector by transforming to a new, smaller set of dimensions that captures primarily the variation in a given training set. Reducing the dimensionality of the input vector obviously also reduces the complexity of the Bayesian classifier, which allows for improved parameter estimation as parameter estimation is can be very difficult in high-dimensionality. Furthermore, PCA provides a way to suppress noise in images as it only captures higher degrees of variation and ignores smaller variations. Most modern approaches to facial expression recognition use PCA or a technique that is similar.

The paper [16] describes two Probabilistic Reasoning Models (PRM), both using a Bayesian classifier. The PRMs are applied to face recognition, but the technique could be applied on facial expression recognition in a similar manner. First PCA is applied to the original object space, in this case the images in the training set. The advantage is twofold as it not only reduces the dimensionality and consequently simplifies parameter estimation, but the smaller set of dimensions can be seen as a set of orthogonal axis which model all variation, which means the classifier will be fairly robust for the training and testing set. However, PCA is not without its drawbacks. The implicit danger is that PCA mistakes small variations of relevant features for input noise or mistakes irrelevant variation such as variation in illumination for a principal component. Apparently this is the price to be paid for reduced dimensionality.

After applying PCA, the Maximum A Posteriori (MAP) rule is used to classify the images. The MAP rule classifies an input vector (image) to the class which returns the highest posterior probability. In order to estimate the posterior probabilities, the conditional probability distribution for each class needs to be estimated. As it is not possible to estimate a conditional probability density function when the number of samples is limited, the density function is modeled to a normal distribution. The problem is now to estimate the correct parameters for the normal distribution, as a mean value and variance are required. The paper [16] proposes two different ways as to estimate the variance. As the input and the sample mean are vectors, and the variance is represented by a covariance matrix. The first model (PRM-1) uses a the same covariance matrix for each class in which is a diagonal matrix containing the sampled variance of each one-dimensional principal component subspace (the variation along each axis). The second model (PRM-2) estimates the covariance matrix based on al the within class scatters in the reduced PCA subspaces.

The results are promising when the performance of the two proposed models are compared to two other methods. The other methods are: PCA without the assumption of a normal distribution and parameter estimation and the Fisherface approach which combines PCA and Fisher's Linear Discriminant. The next step is to make selection of PCA subspaces rather then just selecting the subspaces with the highest variation. Furthermore, special weighting matrices may be used to amplify the variation in a particular subspace to improve recognition. Alternatively Probabilistic Principal Component Analysis might be used such as described in [16].

3.7 Hybrid Bayes in FER

Instead of using a Naïve Bayesian classifier such as in [12], [17] proposes an alternative approach for facial expression recognition by using a hybrid Bayesian network. A Hybrid Bayesian network (HBN) is a Bayesian network which consists of both discrete and continuous variables. The inputs (observed facial features) are several specific areas of the face, rather than a set of key points spanning the entire face. In [10] the face is decomposed into four components: face shape, right eye shape, left eye shape and the mouth shape. The motivation for this decomposition is the belief that the shape of the face is individually independent and decomposition can be efficiently used to capture the manifolds of the facial expressions. So instead of defining

facial expressions as holistic entities, they are represented by a combination of intrinsic functionalities of the following subcomponents:

$$\text{Expression} = \text{state}_{\text{mouth}} + \text{state}_{\text{eyeRight}} + \text{state}_{\text{eyeLeft}}$$

The claimed merits of this approach are the intuitive approach and the ability to classify similar expressions without any additional overhead (e.g. smile with eyes closed, smile with eyes open). Furthermore, performance with the proposed hybrid Bayesian network is expected to be better than Naïve Bayes, since Naïve Bayes assumes independency of the observed features given the classified expression. However, dependency between different features can be extremely useful, particularly in cases when the input images are partly occluded and not all features are visible. To incorporate dependency in the hybrid network (figure 3.2), hidden variables are included that model the dependency between the left eye, $P(EL1|LR)$, and the right eye, $P(ER1|LR)$. The same is done for the corners of the mouth. In figure 3.2 round nodes are continuous and square nodes are discrete. The unshaded nodes are hidden.

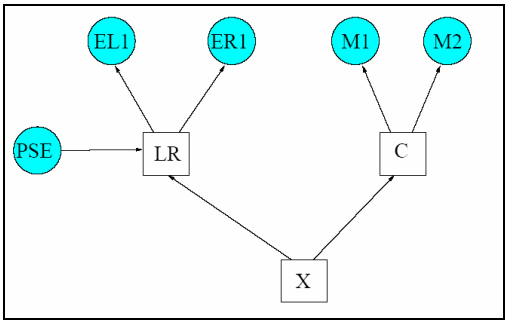

Figure 3.2: Hybrid Bayesian network

In the experiment described in [17] the Active Appearance Model (AAM) (Bettinger et al. 2002) is used the make a mean representation of the observed face. Furthermore a pose estimator is used in conjunction with Probabilistic PCA (PPCA) (Tipping and Bishop 1998). Probabilistic PCA is better suited as normal PCA as it has been found that it provides a much more accurate estimation of the principal components by finding the relationship between projected angles and parameters of the observed face. The model operates by first detecting the face in a newly observed video stream. Then the observed face is transformed to a frontal view representation using the pose estimator. The face is then decomposed into the specified subcomponents and the subcomponents are fed to the hybrid Bayesian networks as input vectors. Finally the avatar is animated according to the observed output. A few examples are shown in figure 3.3. To test the performance of the hybrid Bayesian network it is compared with a Tree-Augmented Naïve Bayesian (TAN) network. Both networks are fed the same image sequences. The result shows that the hybrid Bayesian network performs slightly better as it reaches an accuracy of 88% versus an accuracy of 82% reached by the TAN. The paper concludes by proposing that the next step in this line of research is to incorporate temporal information into the model by using Dynamic Bayesian Networks.

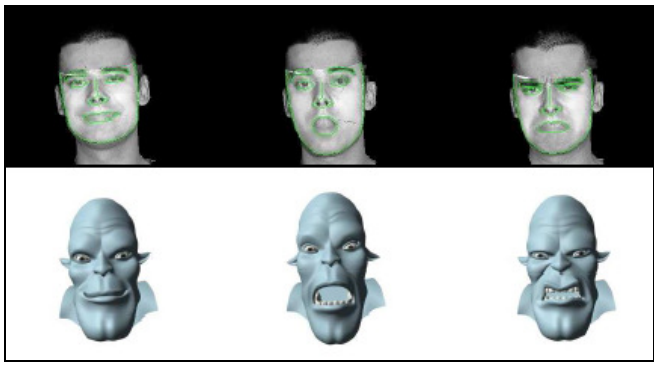

Figure 3.3: Animated avatar on the output of the HBN

3.8 Dynamic Bayesian networks and affect detection

In the paper [18] D. Datcu and L.J.M. Rothkrantz propose an automatic system for facial recognition in video sequences. It makes use of both a Bayesian network and the FACS model. The first step in the recognition process is to locate the eyes of the face in the video stream. This is accomplished by an eye-tracking mechanism based on a Kalman filter (Kalman 1960). After the position of the eyes are determined, the rest of the visual features is recovered. The next step is to place key points on the observed face, which are 30 specifically located points on the face as described by the Kobayashi and Hara model (fig. 3.5). These key points are then used to determine the visual facial parameters. In turn, these visual parameters are then mapped on a number of corresponding AUs.

This is where the Bayesian network is applied. The network is modeled to contain knowledge about the relations between certain Action Units and the related labels for the expressions, which are the seven prototype expressions. The probability of each facial expression is then determined separately. Each facial expression is dependent on the presence of a number of specific Action Units or specific combinations of Action Units. For example, the expression 'happy' primarily depends on the presence of AU 12 or the combination AU 12 + AU 6. In turn, AU 12 and AU 6 are dependent on the facial parameters based on key points 2, 6, 8, 15, 16, 17 and 21. In addition to the observed AUs, the probability of each facial expression in the video stream is also dependent on the expression observed directly before the current expression. This is because expressions are assumed to change 'smoothly' over the course of time and not within a matter of frames. Furthermore, the Bayesian network also takes into account dynamical changes of the observed key points. After the probability of each facial expression is determined, the expression with the highest probability is then elected as the result of the system.

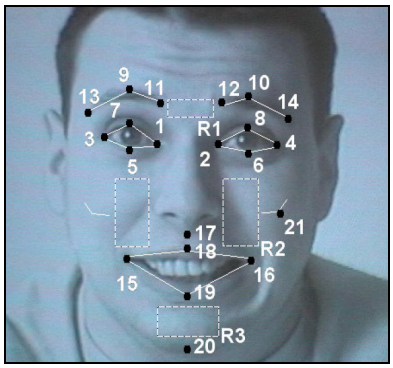

Figure 3.4: Facial key points

The conditional probability tables are determined offline by images from the Cohn-Kanade AU-Coded Facial Expression Database. The system is able to operate real-time, which is a hard demand for most applications of facial expression recognition. Although [18] does not contain quantative performance results, the concluding remark states that the results are very promising. Furthermore, the proposed Bayesian network incorporates dynamical data. It uses the anterior facial expression and key points to estimate the current facial expression. Although not explicitly stated in [18], in this sense the proposed Bayesian network could be interpreted as a Dynamical Bayesian network.

The facial expression is not the only aspect of a human that betrays his or her emotional state. Physical behavior can also give a good indication of a persons mental state. For example, rapid movement of the limbs might indicate a state of agitation and a the closing of the eyelids for specific lengths of time might indicate fatigue. Here we will briefly discuss two papers which deal with emotion detection without focusing exclusively on the human facial expression. Furthermore, both papers achieve the emotional state detection by using Dynamic Bayesian networks. A Dynamic Bayesian network (DNB) is a Bayesian Network which incorporates data over different time segments instead of just a single moment, i.e. that if a specific state has been detected at timeframe *t-1*, the same state will occur with a heightened probability at timeframe *t*. Currently (to the best of the authors knowledge) there has been no research specifically focused on facial expression recognition using Dynamic Bayesian networks with the possible exception of [18], even though enough research has been done on spatio-temporal analysis of facial expressions in the past decade [10, 11].

The paper [19] describes a system that uses a Recurrent Bayesian network to detect violent human behavior. A Recurrent Bayesian network is a particular Dynamic Bayesian network which incorporates not only the previous state of the network, but the entire series of previous states. The system is used to detect violent behavior in a group of people and violent behavior is defined as a high level of agitation in a group. Instead of using facial features, such as in facial expression recognition, the pixels which constitute the observed group are monitored for changes in acceleration, density, size and direction. The Recurrent Bayesian network is trained with 600 video streams and tested with 10 streams of the same site. Results are promising as all test samples were correctly classified. However, there are still some problems, e.g. violent behavior is only detected when an enduring state of agitation is observed, and therefore violent behavior is detected with a delay. On the other hand, violent behavior that is restricted to a very short time window may not be detected because of the dependencies between time segments.

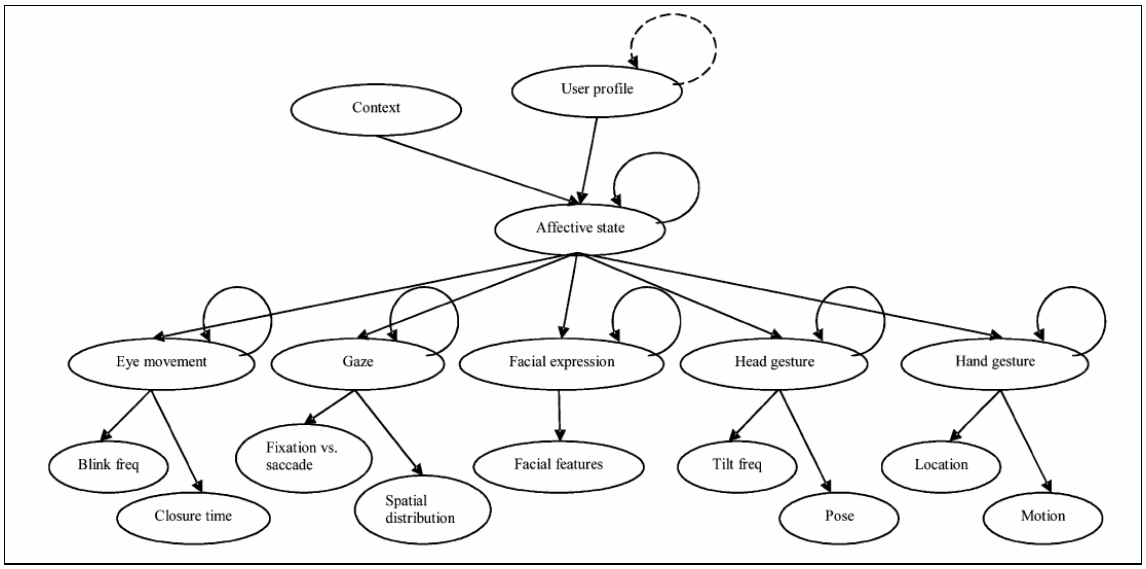

Figure 3.5: A general model which shows how a static Bayesian network can be transformed to a DBN by adding temporal links.

The paper [20] describes an automated agent that is able to detect the affective state of a user and provide assistance accordingly. A nice example is Microsoft's 'Office Assistant', which incidentally also makes use of a Bayesian network. The agent proposed in this paper makes use of a Dynamic Bayesian network, as the detection of the current affective state (neutral, confused, fatigued or nervous) is dependent of both the hidden variables and affective state in the previous time segment, and the observed features in the current time segment. The observed 'features' of the user are detected by multiple sensors, such as tracking of the users gaze, eyelid movement and facial expression. Figure 3.5 shows the general model proposed in [20]. The observed features are placed in the bottom layer. The hidden variables are modeled directly on top of the bottom layer. The affective state is dependant on the hidden variables and two internal variables, which represent the complexity of the current task and the modeled user profile. Straight arrows indicate static dependencies, looped arrows indicate temporal dependencies.

In [20], determining the emotional state of the user is not the end-goal. The current affective state is used to estimate whether or not the agent should intervene and, if so, what type of assistance is required. For this reason the model also keeps track of the user profile and the complexity of the current task. A user that is highly intolerant for intervention from an automated agent is less likely to be surprised by intervention of the agent. A user performing a task that is relatively complex is more likely to benefit from intervention. To decide which type of assistance is most likely to be needed (warning, emphasis or simplification) the benefit and cost for each type of intervention are compared, taking into account the probabilities of the possible affective states of the user. Several experiments with both synthesized and real data are described, with the intention of proving that such a model would be feasible. Application of this type of agent is not restricted to a specific area as it can be applied to areas such as office related software, where fatigue is a common occurrence, or army related control software where people have to operate under highly stressful conditions. Although the general model in figure 3.6 includes facial expression analysis, the paper [20] relies more heavily on other sensors to operate and does not elaborate on this possibility.

4 The Small Sample Case

Using Bayes as a classifier also has some considerable drawbacks. For example, if we want to apply a Bayesian network to a problem we have to model the prior probabilities. Without priors Bayes is not much use. So we try to estimate the prior distribution based on the available training data. Now it is often said that training data must span the entire domain in order for us to make a good estimate of the true distribution. Unfortunately, it is quite exceptional for researchers to have that amount of data. The consequence is that we make poor estimates of the prior distribution and consequently make poor Bayesian classifiers.

4.1 The curse of dimensionality

This phenomenon is known as the curse of dimensionality: the amount of features that we need to extract per data sample is significantly greater than the amount if available training samples. This particularly affects Bayesian classifiers as they depend on a probability distribution which needs to be estimated from the available data. On the bright side, attempts have been made to make up for the curse of dimensionality. This chapter discusses three ways to overcome this problem: training with unlabeled data, generating new data with facial expression synthesis and using large-margin classifiers instead of Bayesian networks.

4.2 Bayes with unlabeled data

We distinguish to different kinds of data: labeled and unlabeled. Usually when we claim there is not enough data available we mean not enough data of the labeled kind. In this case labeled data are samples which have been preprocessed, for instance the data might have been coded by certified FACS coders. The culprit is that labeling data requires time, expertise and training. On the other hand, unlabeled data is readily available as it is quite easy to collect. All that is needed is a camera and a number of subjects. It has been shown in the literature that using labeled data to train classifiers achieves considerably better results.

The paper [21] proposes a way to train a Bayesian classifier with a relatively small amount of labeled data in combination with a large amount of unlabeled data. The first step is to train the classifier with the small amount of labeled data and supervised learning techniques. The next step is to improve the obtained classifier with the unlabeled data. In previous attempts this has usually resulted in a deterioration of the classifiers. In [21] it is argued that the reason that training with unlabeled data results in worse classifiers, is due to the poor modeling of the classifiers themselves. It is argued that if a Bayesian network has the correct parameters and the correct structure it can be an optimal classifier as it accurately represents the posterior probability distribution. A way to search for the optimal structure is to switch structure when performance degradation is detected during training.

In order to solve this problem an algorithm is proposed to find the optimal structure for a Bayesian classifier focused explicitly on improved classification. The algorithm is known as the Stochastic Structure Search (SSS). It functions by randomly selecting a structure and then either adding, removing or changing the direction of a single connection in the network. This method is known as the Monte Carlo Markov Chain technique. If the expected error of the new network is smaller than the error in the current network, the new network is accepted with a high probability. This process is iterated and uses simulated annealing, meaning that the probability of accepting a new network is artificially is reduced as the number of iterations grows in order to ensure convergence. The resulting network is then used as a classifier trainable by both labeled and unlabeled data.

In order to measure performance of the network generated by the Stochastic Structure Search it is compared with two other common network structures: Naïve Bayes (NB) and Tree-Augmented Naïve Bayes (TAN). A Tree-Augmented Naïve Bayesian network is similar to a Naïve Bayesian network in that all features are children of the class label, but in addition all feature nodes may have a single other feature node as a parent. This results in a particular kind of tree structure. Both of these network structures are known to have degrading performance when trained with unlabeled data.

Table 4.1 shows results for both samples from the Cohn-Kanade database and the Chen-Huang database. It shows the performance for each network in case of training with only labeled data and in the case of training with both labeled and unlabeled data. It can be seen that performance degrades for both the BN and the TAN as unlabeled data is added to the

training set. It also shows that the SSS network, trained on both labeled and unlabeled data, has slightly better performance than the BN and TAN networks even though they have been trained exclusively on labeled data. The paper concludes by stating that if the SSS algorithm is applied with only labeled data even better performance is expected, which proves the worth of labeled data.

Table 4.1: Results of SSS in comparison to NB and TAN

Dataset	# Samples labeled	# Samples unlabeled	# Test	NB labeled	NB labeled & unlabeled	TAN labeled	TAN labeled & unlabeled	SSS labeled & unlabeled
Cohn-Kanade	200	2980	1000	72.50%	69.10%	72.90%	69.30%	74.80%
Chen-Huang	300	11982	3555	71.25%	58.54%	72.45%	62.87%	74.99%

4.3 Synthesis of facial expressions

Another possible solution to the small sample problem is to find a way to generate new samples. One area of research that is currently blooming is that of face expression synthesis. The idea is that we need only a neutral image of a subject in order to generate a new image of the subject with an arbitrary facial expression. Different approaches have been tried, such as creating a 3D wire frame model that captures the possible motions and restrictions of a face. But if we are to use generate data for training, we will need images that have a high fidelity with real-life images, which is often not the case with 3D model generated data. In the publications [22] and [23] a computational model for facial expression is proposed, which generates a mapping from an image of face showing a neutral expression to the same face showing an arbitrary expression without regard to sex or skin color of the subject. This is achieved by using two models in conjunction with each other: the Facial Expression Shape Model (FESM) and the Facial Expression Texture Model (FETM).

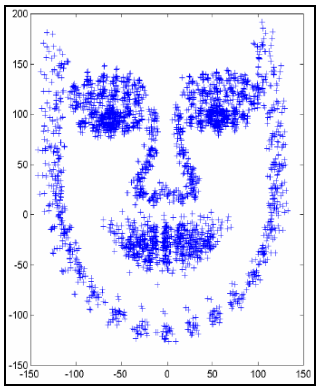

Figure 4.2: Alignment of the landmark points

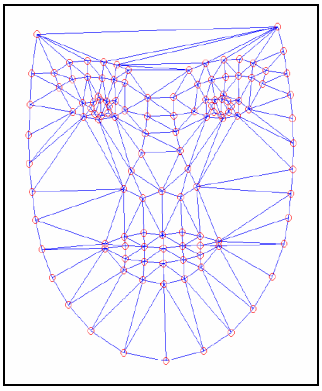

Figure 4.3: Delaunay Triangulation

The Facial Expression Shape Model is a statistical model based on point distribution that only allows for deformations observed in the training set, and consequently deformations of the human face. The training set that is needed to train the FETM is labeled with the help of the Facial Action Coding System. Each FETM is build for a specific expression, or in other words a specific set of Action Units. The FESM is created by first manually marking the images in the training set with 122 specific landmark points. Furthermore, these landmark points are weighted according to their level of importance depending on the AUs that the FETM is build for. For example, if we want to make a mapping for AU 12 we would pay specific attention to the corners of the mouth.

In order to analyze the variation in the location of these landmark points, the faces in the training set need to be aligned as closely as possible. This is achieved by a technique known as generalized Procrustes alignment (GPA). This technique aligns two shapes with respect to position, rotation and scale by minimizing the weighted sum of the squared distances between the corresponding landmark points. After GPA, the training set is analyzed with Principal Component Analysis to lower the dimensionality of the feature input space and to capture the variation of the landmark points. The Facial Expression Texture Model is created in a similar manner. All images are warped to a mean shape and then Delaunay triangulation is used to create

a wire frame of 214 separate regions between the landmark points. Each face is then converted from color to greyscale and pixel values in the triangles are transformed to an input vector. Principal Component Analysis is also used here to reduce dimensionality and capture variance of the input vectors. To create a mapping from the neutral expression to a selected set of AUs, different approaches are proposed such as the used of a Feedforward Hetero Associative Memory Network (FHMN), Radial Basis Functions (RBF) and Support Vector Machines (SVM). Support Vector Machines and Radial Basis Functions will be briefly discussed in the next paragraph. In the experiment described in [23] a Feedforward Hetero Associative Memory Network is used to model the FESM and Radial Based Functions are used to create a mapping for the FETM.

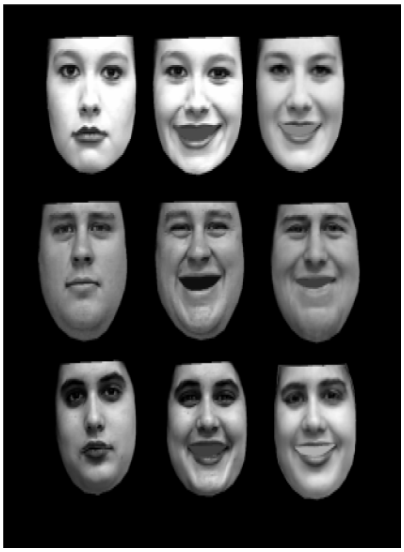

Figure 4.4: The first column shows the real neutral images, the second column shows the real 'happy' images and the third column shows the synthesized 'happy' images. None of the faces were part of the training set.

The experiment describes a FESM and FETM trained to warp a neutral face to a face showing Action Units 6, 12 and 15. This combination of AUs is generally agreed upon to depict a 'happy' expression. Forty people and eighty images from the Cohn-Kanade AU-coded facial expression database were used to create a training set. The results of the experiment are visible in figure 4.4. Numerical results are measured by calculation of a correlation coefficient between the real 'happy' images and the synthesized 'happy' images for both the FESM and the FETM. The FESM achieved an average correlation of 0.814 and the FETM achieved an average of 0.7799. It should be noted that this experiment also used labeled data in the training set.

Although [23] shows promising results concerning the synthesis of photo-realistic facial expressions, it should be duly noted that it does not suggest the use of synthetic images for training purposes or in other words as the solution for the curse of dimensionality. It simply provides a way to transform unlabeled data consisting of images of neutral faces to labeled data consisting of faces portraying a set of specific Action Units. This in essence is a way to handle the curse of dimensionality as generating unlabeled data can be done relatively effortless and without regard to sex and skin color of the subjects. However, before the assumption can be made that labeled synthesized data can be used to train classifiers such as Bayesian networks, this possibility shall have to be researched and tested.

Another paper [24] proposes a different method to achieve facial expression synthesis. Instead of creating a separate model for the synthesis of each expression a single model is being created that can generate each of the seven expression for an unknown person. In order to get an idea of this different approach we will discuss it briefly.

Given an corpus of facial expression images of different persons, an attempt is made to decompose the data into two separate subspaces: the expression subspace and the person subspace. High-Order Singular Value Decomposition is used to decompose the tensor *A* which is a third-order tensor (number of facial expressions, number of persons and number of salient features):

$$A = S \times U^{person} \times U^{expression} \times U^{feature},$$

in which *S* is the core tensor representing the inter actions between the decomposed subspaces. By separating the knowledge of persons and expressions, it is possible to generate new expressions for all known persons in U^{person} given the new expression vector u^e, but far more interesting is to ability to generate all known expressions in $U^{expression}$ if we have a new person vector u^p.

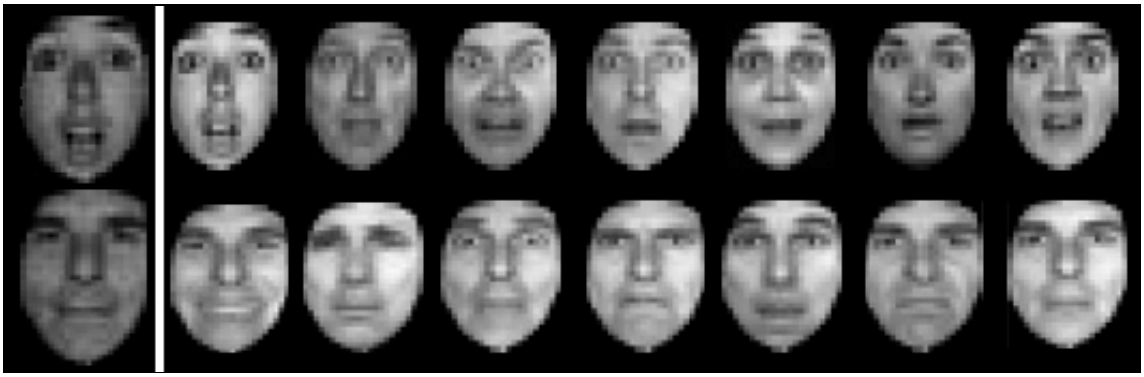

Figure 4.5: Generating a new expression for all known people (first row) & generating all known expressions for a new person (second row). The first column shows the new expression/person.

Figure 4.5 shows some of the results from [24]. The first row shows a particular synthesized expression for all known persons, given the new unknown (portrayed) expression on a known person. The person originally portraying the new expression has to be known in order to know the correct mapping for the other persons. The second row shows the synthesis of the seven expressions of joy, sadness, fear, anger, surprise, disgust and neutrality, given an unknown person with a known expression. Although the results are not discouraging the created model is not very robust to distinctive facial features such as beards and mustaches and performs relatively poorly if the person subspace does not contain a person that is similar to a arbitrary new person.

4.4 Other classification approaches

The very recent paper [25] discusses the problem of facial recognition in the small sample case. A novel approach, named Feature Selection via Linear Programming (FSLP), is proposed and compared with three other often used methods: Bayes, Support Vector Machines and AdaBoost. The paper argues that the first problem in facial expression recognition is feature selection, as there are often a large number of features to choose from. This is the process of selecting the distinguishing features for the data in the training set, before actually training. Unfortunately there is no exact way to determine the optimal set of features. Although there are algorithms available to calculate an optimal set of features, such as the Sequential Forward Floating Selection by Pudid *et al.* , these are very complex and time-consuming. Both Bayes and AdaBoost use a heuristic approach and SVM often uses all available features.

A novel approach is proposed that addresses the problem of feature selection during training instead of before. The algorithm works by finding a function such that $f(x)>0$ if x is an element of *A* and $f(x)<0$ if x is an element of *B*. Finding the function for this binary classification problem is achieved by linear programming. In addition the algorithm has the property of being a large margin classifier and is therefore somewhat robust to the curse of dimensionality. When a classifier is created with example-based learning, as is the case with Bayesian networks and particular artificial neural networks, the discriminate function might result in separating hyperplanes which are biased towards the last sample used for training. Figure 4.5a shows several such hyperplanes, all of which are qualified to classify the training set, but are likely to perform poorly on unseen data. This problem is

likely to occur in the small sample case, as there are simply not enough samples to converge to an optimal discriminate hyperplane.

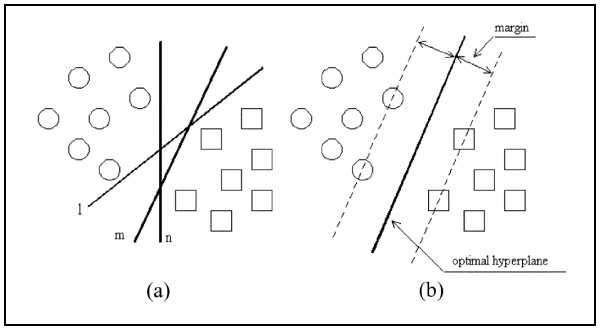

Figure 4.5: Large margin classifiers (b) compared to other classifiers (a).

Large margin classifiers, however, are still able to find an optimal hyperplane by finding the hyperplane that provides the greatest distance between the hyperplane and the nearest data point in each class such as shown in figure 4.5b. A larger margin allows for better generalization and consequently better performance when we are faced with a limited amount of training samples. Both SVM and AdaBoost, as well as the LP approach qualify as large margin classifiers, but Bayesian networks, on the other hand, do not.

The paper [25] also argues that the multiclass classification problem can be simplified. Usually, each class is compared to all others in order to determine its probability. Hence the features of each class are chosen to achieve distinction from all other classes. However, this is not an optimal approach as, different features are used to distinguish different pairs of classes. For instance, the features that are useful to distinguish the letter 'E' from 'F' may differ from those features needed to distinguish 'E' from 'L'. In facial expression recognition we may find that the mouth is of key importance when distinguishing 'happy' from 'sad' and the eyebrows are more important when we distinguish 'sad' from 'surprised'. So a better strategy is to make pairs for each two separate classes and assign a subset of distinguishing features for each pair. Thus for a classification problem with 7 classes (prototype facial expressions including neutral) we would end up with 21 pairs. The most probable class can then be found by determining the class which is most probable in most pairs.

Table 4.6: Comparison of results

	Bayes All	Bayes FS	AdaBoost	L-SVM	NL-SVM	FSLP
Accuracy	63.3%	71.0%	71.9%	92.4%	91.9%	91%
# Features	612	60	80	612	612	17.1

In order to test the proposed results a database of 213 images of 10 Japanese women was used. The results can be found in figure 4.6. The six methods compared are: a Bayesian network without feature selection (Bayes All), a Bayesian network with pairwise feature selection (Bayes FS), AdaBoost, Linear Support Vector Machines (L-SVM), Nonlinear Support Vector Machines (NL-SVM) and Feature Selection via Linear Programming (FSLP). Obviously Bayes and AdaBoost perform relatively poorly when compared to Support Vector Machines and FSLP. The advantage of FSLP over SVM is that it is less costly from a computational point of view.

The paper [25] concludes by implying that Bayes cannot solve the small sample case problem, as it is outperformed by all other methods. The reason is that, for a Bayesian classifier, we need to estimate the conditional probability distribution for each class. In the small sample case it is hard to span these distributions and we end up with biased and deformed distributions.

5 Conclusions and recommendations

Research on facial expression recognition is still an open chapter, meaning there is not yet a strong convergence towards a single method or technique which outperforms all the others. A large variety of methods have been developed, tested and improved [11]. Some of those techniques have earned their stripes such as Principal Component Analysis (PCA) and the use of the Facial Action Coding System [7]. Furthermore, great efforts are being made towards conformity of experiment results, such as the implementation of the Cohn-Kanade database [9].

Other techniques are discarded as new ideas are applied and found to be more effective. Although there are a handful of coding systems available for describing facial expressions, only FACS is still in wide spread use. The question that we ask ourselves is: is Bayes a deprecated method when it comes to facial expression recognition.

5.1 Problems in facial expression recognition

Many of the problems that vex facial expression recognition are well known, but have yet to solved efficiently. This is also the main criticism against the current models. Many of the models discussed in this survey still perform poorly when confronted with input images that suffer from poor illumination, varying pose and partial occlusion of the face. None of the models proposed in the literature encompassed by this survey introduce explicit solutions to these problems with the exception of [17] where recognition is possible from varying angles and recognition with partial occlusion is still possible. Almost all current models assume anterior knowledge of the position of the face, good lighting conditions and full visibility of the face. These assumptions result in poor performance when applied to real-life situations. It should be noted that these problems lie in extracting facial key points and not so much in analysis of the observed key points. In other words, these are not problems that are to be solved by a Bayesian network.

However, analysis of the observed face brings its own set of problems. Most of the models we have discussed assume a very simplistic approach towards modeling facial expression. Naive Bayes and Tree-Augmented Bayesian networks have been tried in the past but these models are known to be inaccurate. The reason for their application is their relatively small amount of needed samples and parameter estimation. It is very uncommon to find a model that is structures in a more complex and accurate way. The paper [21] shows a novel approach by modeling a network by examining the dependencies in the data, instead of designing the model and then fitting the data to it. The paper [18] also introduces a more complex but accurate model by incorporating the Action Units from FACS explicitly in the network.

Another trend that is to be expected is the rise of multimodal models. Almost all of the models discussed in this survey use only a single modality: observation of the face. The reason humans perform so well in recognizing emotional context is because the we are able to combine multimodal information. For example, we do not only observe the facial expression, but we are also able to pick up emotion from the stress level of someone's voice or from someone's body language. Some of the papers discussed in this survey suggest a multimodal approach as the next step in their research. However, this survey has not focused in particular on determining the emotional state of a subject in a multimodal manner, so the lack of multimodality is an observation rather than a hard point of criticism.

A final point of attention is the lack of conformity that exists between the testing data of all proposed models. Even after the introduction of the Cohn-Kanade database [9], many (not all) of the proposed models are tested and trained on customized training sets or only a small part of the Cohn-Kanade database. This makes it difficult to compare models and thus to discern which models actually yield the best results. Furthermore, although it is the largest image resource available, the Cohn-Kanade database is not yet complete as many of the frames in the video sequences are yet to be AU-coded.

5.2 Bayesian networks on FER: state of the art

We may ask which is the best model that applied to facial expression recognition using Bayesian networks. Comparing the results of different models is difficult because the difference in testing sets. However, It is possible to say something about which elements increase performance and which techniques have proven to be inefficient.

The state of the art in FER using Bayes is a combination of the models described in [17, 18 & 21]. The model proposed in [17] provides a model that is to some extent robust against varying poses of the observed subject and partial occlusion of the face. This is achieved by using an Active Appearance Model (AAM) and a pose estimator. Partial occlusion is solved by assuming specific dependencies between certain facial features. For example, if the right eye is occluded, the information retrieved from the left eye makes up for this, still allowing correct classification. The paper [21] shows the state of the art when it comes to structuring a model that fits to the data by using a Stochastic Structure Search algorithm. Most current models work the other way around, fitting data to an assumed model, which essentially results in a performance that is dependent on the correctness of the model. One of the prime benefits of fitting the model to the data is the ability to train the model on unlabeled data. This provides us with a way to counter the small sample case. Finally, the paper [18] shows a model that explicitly incorporates temporal information when analyzing which expression is observed. Temporal information can be very helpful in classifying expressions in video streams as it has been shown that there are strong temporal dependencies the separate frames of a video sequence.

All of the models proposed in [17, 18 & 21] use Principal Component Analysis to capture variance in the input images in order to achieve better classification results. The next step for PCA is to determine which principal components capture the most discriminating variance. Current methods for PCA usually just use the components which cover most of the variation. The problem is that in this way image noise and lighting conditions are assumed to be principal components, when it is desirable to only capture the variance that is present between faces and expressions.

A final technique that is currently performing nicely is pair-wise feature selection as proposed in [25]. In a multi-class classification problem it is highly inefficient to compare all classes by a fixed set of facial features, as different subsets of features are the discriminators for each class. The proposed idea is to compare each combination of two classes, as each pair only has a small set of discriminating features. To determine the most likely class, we could then select the class that most probable most often (of all pairs). Furthermore, this approach allows us to better discriminate classes that are very alike, such as fear and surprise, because we focus in particular on the discriminating features.

5.3 Tools

The papers discussed in this survey do not make use of specific software tools intended for facial expression recognition. All of the discussed models have been self-implemented. However, some tools are available. The Integrated System for Facial Expression Recognition (ISFER) developed by Pantic & Rothkrantz (2000) is such a tool that automatically classifies a frontal static image to an emotional label. The inference engine that enables this is named HERCULES. The notable thing about HERCULES is that it consist of a number of separate reasoning modules that are assigned to specific parts of the observed face. The motivating thought is that each technique is has its own particular application in which it performs best. For example, a neural network is used to classify the state of the eyes and a fuzzy classifier is used to classify the state of the mouth. Although the HERCULES inference engine can easily be expanded with new models, due to its architecture, to the best of the authors knowledge, Bayesian Belief network modules have not been included yet

One of the most popular non-software tools for facial expression classification is the Facial Action Coding System [7]. A new version of FACS was published in 2002, refining the older version dating from 1978. FACS is still in widespread use today, as it is regarded as a comprehensive way to code all possible facial expressions, be it voluntary or spontaneous [8]. Most of the classifiers introduced in this survey perform considerably better when trained on data that has been pre-processed by use of FACS or some other coding scheme. The probable cause for this increase in performance can be found in the fact models are more apt recognizing faces in a analytical manner (i.e. recognizing Action Units) then in a holistic manner (i.e. recognizing the expression as a whole).

Another useful ‘tool’ is the Cohn-Kanade database [9] which has been extensively discussed in previous section and in particular in section 3.3. It provides the research community with a uniform way to measure performance of the proposed systems. Although widely known, the Cohn-Kanade database is not exploited as often as could be expected from its apparent

usefulness. The database itself is still under construction, as not all of the images and frames have been AU-coded, but since the introduction of the database in 2000 no notable improvements have been reported.

Although most of the models discussed in this system use their own software modules, capable tools for modeling Bayesian networks can be found in SMILE and GeNIe [26], developed at the Decisions Laboratory, University of Pittsburgh. SMILE is a fully portable library implementing graphical decision-theoretic methods such as Bayesian networks. GeNIe is a development environment for graphical decision-theoretic models, making use of the SMILE library. Both SMILE and GeNIe are still under development.

5.4 Current research

The current research performed by Yacoob et al. [10] is focused on facial recognition and physical action detection rather then facial expression recognition. However, analysis of human gestures might very well help analyze the affective state of a subject, which makes his research akin to that of facial expression analysis. Cohen et al. [14, 15, 21] is currently researching the possibilities for learning the optimal structure for Bayesian networks in order to learn from unlabeled data such as proposed in [21]. This might prove a feasible solution for the small sample case. Cohn et al. [9] is currently working on ways to improve Active Appearance models. He proposed a method to model an AAM from multiple views of a single face, regardless of the position and geometry of the cameras shooting the face. Rothkrantz et al. [11, 13, 18] most recent work on facial expression recognition involves exploring other methods for facial expression analysis besides Bayesian networks. Although the application of Bayesian networks to facial expression recognition has been examined [18], other classification techniques outperform Bayes in the common small sample case [13]. Ghent et al. [22, 23] is currently working on photo-realistic facial expression synthesis. Most models that represent the human face are 3D wireframe models that are able to imitate and synthesize facial expressions, but not many of them are photo-realistic. Photo-realistic facial expression synthesis might provide us with a feasible solution to the small sample case. Gong et al. [17] is currently preparing for the IEEE International Workshop on Analysis and Modeling of Faces and Gestures. This also involves exploring novel techniques and applications for facial expression analysis. His research is not strictly limited to Bayesian networks or facial expression recognition.

5.5 Conclusions

One of the biggest problems in facial expression recognition is the lack of training data. Although efforts are made to counter this problem, most research is forced to deal with a small sample case. In theory, a Bayesian network is an optimal classifier as it accurately represents the conditional probability distribution of any given problem, under the condition that it is properly trained and modeled. This means that we have to make an accurate estimation for the local probability distributions. Obviously, the quality of such an estimation depends on the available data. In order to model an accurate distribution this data has to contain samples that span the entire distribution. This is were Bayes trips and falls over, as it there is rarely enough data to span the entire distribution and frequently not even enough data to make a decent estimation. The consequence is that a Bayesian networks fails to accurately represent the conditional probability distribution and consequently has a deteriorated performance

To counter this problem several possible solutions have been explored. The obvious solution is to produce more data. Although it is not a problem to acquire more images of facial expressions, it quite an undertaking to label, encode and structure these images. Therefore, methods have been explored that learn from unlabeled data [21]. Furthermore, it might be possible to synthesize usable data such as described in [22, 23 & 24]. The question remains if these are feasible approaches, as more research is needed before this can be determined. Furthermore, other methods have also been tested in the small sample case and have proven to be more useful. Support Vector Machines (SVM), Relevant Vector Machines (RVM) and Radial Based Functions (RBF) are some of those methods. The main reason these so called ‘large margin classifiers’ outperform Bayes is because they make no attempt to model a conditional probability distribution, but simply use the small amount of available data to estimate the optimal discrimination functions between the different classes.

Although it is hard to compare different methods, as most of them are tested on different sample sets, it seems that Bayesian classifiers have a diminishing viability. Naïve Bayes has already proven to be inadequate for facial expression recognition, as the assumed conditional independence is an unfounded assumption when applied to the features of the human face. In fact,

dependency amongst features if often very useful as it allows facial expression analysis even when parts of the face are occluded [17]. The reason that Naïve Bayes is still used frequently is because the drawback that it is often a bad fit to the underlying model is offset by the ability to train on limited data. Tree-Augmented Bayesian networks have also been outperformed by other proposed models on the same training data. Research has been done with Bayesian networks that better fit to the available data with losing its generalizing ability [17] such as Hybrid Bayesian networks or Bayesian networks with optimized structures [16].

The future of the Bayesian approach lies in the ability to incorporate temporal data. Dynamic Bayesian Networks (DNB) is might provide a way to improve classification of facial expressions, but DNBs are a relatively new area of research and (to the best knowledge of the author) no pure DNB approach has yet been applied to the problem of facial expression recognition yet, although [18] uses dynamical variables. The current area of application for DBNs is speech recognition. However, [19] and [20] demonstrate promising results on applying a Dynamic Bayesian Network on the recognition of specific behavior of humans and deferring their affective state. It seems feasible that a similar approach can be applied to facial expression recognition. The concept of spatio-temporal analysis has been around for about a decade, but, as DNBs are a rather new trend, it has yet to be applied in conjunction with a Dynamic Bayesian network. Currently, analysis on video streams with a static Bayesian network simply estimate the probability for each frame.

It is the authors final impression that static Bayesian networks are a passed station when it comes to facial expression recognition. However, new ways to apply Bayes, such as Dynamical Bayesian networks, might prove that this particular probabilistic model can be an extremely powerful classifier if modeled in the right way.

5.6 Recommendations

Based on the research reviewed in this survey, the author suggests three possibilities for new research involving Bayesian networks and facial expression recognition.

Dynamic Bayesian networks [18, 19] provide us with a way to incorporate temporal data. Facial expressions occur in a specific time frame and it has already been shown that the behavior of the facial features over time are correlated with the observed expressions [10]. Furthermore, static Bayesian networks can be made dynamic by connecting the nodes of separate static Bayesian networks in sequential time steps. The paper [18] proposes a Bayesian network that incorporates temporal data into a static network, but it is not explicitly constructed as a Dynamic Bayesian network. The author proposes a hierarchical Hybrid Bayesian network such as discussed in [17] made dynamic with temporal dependencies such as shown in figure 5.1. The blue arrows indicate temporal dependency. The reason for electing this structure over Naïve Bayes and Tree-Augmented Naïve Bayes is that the Hybrid Bayesian network in [17] contains hidden nodes. Hidden nodes do not only reduce the complexity of the network but when connected in a dynamic way they might provide more nuanced temporal information than when only the diagnostic node is dynamically connected [18] and lead to increased performance compared to static Bayesian networks. NB and TAN have no hidden nodes and therefore could only dynamically connect the diagnostic node.

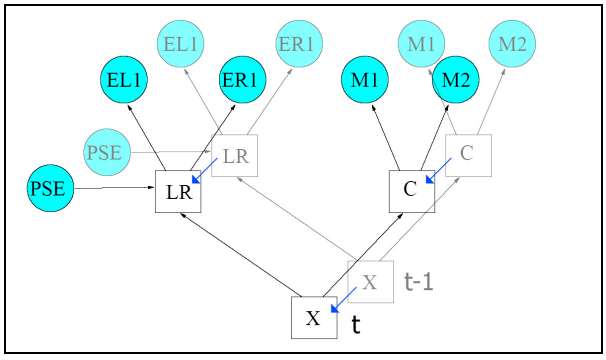

Figure 5.1: proposed dynamic Bayesian network

A second venue of possible interest in examining the viability of using photo-realistic synthesized images to train Bayesian networks. As previously discussed, Bayesian networks suffer from a lack of sufficient training data because its hard if not impossible to estimate the conditional probability distributions. Recent work [21, 22, 23] has examined the possibility of synthesizing an arbitrary set of Action Units from a single image of a face with a neutral expression. The results were promising as the method proved independent of age, sex and race. Although the proposed synthesis method is not specifically meant to solve the small sample case, it is effectively a way to transform unlabeled data into labeled data. The author proposed a static Bayesian network such as described in [12]. The reason for choosing this network structure is because it incorporates the recognition of Action Units. In order to test the viability of synthesized data, two separate instance of the same network would be trained. One network would train on a real data, and the other would train on synthesized date. The synthesized-image data set would be several magnitudes greater than the real-image data set, since it is generally easier to produce. If training with synthesized data proves successful, a system could be considered that integrates synthesis and training. In other words, we would have a model that is able to train on a set of only neutral face images.

A third possible venue of research is using a Bayesian network that does not attempt to learn from static facial features, but tries to recognize dynamical elements in a video stream of a facial expression. In the discussed literature, all Bayesian networks analyze a number of facial features at the same moment, under the belief that the combination of features is a unique key to identifying the corresponding emotional prototype. The author proposes a Bayesian networks that takes only a limited number of features as input, but has several inputs for each feature, each at a different moment in time. The motivation thought is that perhaps emotional prototypes are not only recognizable by a large number of static features, but also by a limited number of features over the course of time. Research has already shown that certain emotional prototypes are identifiable by the behavior of the face over time [10]. Figure 5.2 attempts to clarify this idea.

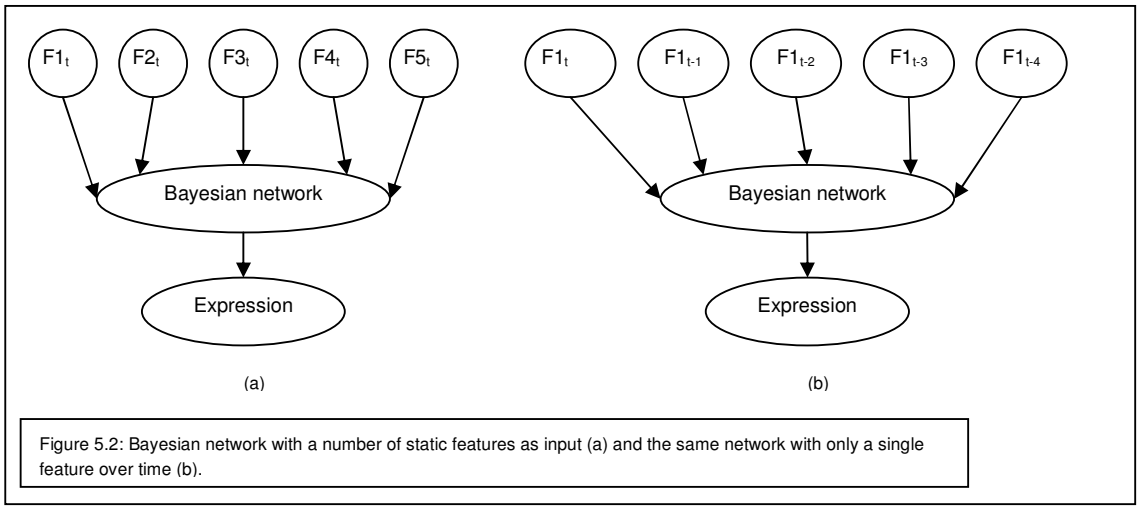

Figure 5.2: Bayesian network with a number of static features as input (a) and the same network with only a single feature overtime (b).

Bayesian networks have proven their worth when trying to model uncertainty. At the end of this survey, the author has the distinct impression that Bayesian networks can be applied to facial expression recognition with great success, but it remains up to us to find a way to apply them so that they can prove their worth.

5.7 Paper overview

Table 5.3 shows an overview of the papers and literature encompassed by this survey and the specific topics that were included per reference.

Figure 5.3: Overview of included literature

Paper reference	Static Bayesian networks	Dynamic Bayesian networks	Other classifiers	Facial expression recognition	Facial Action Coding System	Data: static images	Data: video streams	Facial expression synthesis	FER History
3	x								
4	x								
5	x								
6				x	x				x
7				x	x				
8				x	x		x		
9				x	x	x	x		
10			x	x			x		
11			x	x	x				x
12	x			x			x		
13	x		x	x		x			
14	x			x			x		
15	x		x	x		x	x		
16	x		x			x			
17	x			x		x			
18	x			x	x		x		
19		x					x		
20		x					x		
21	x			x	x	x			
22				x	x	x		x	
23				x	x	x		x	
24				x		x		x	
25	x		x	x		x			

6 References

[1] R. A. Zimmerman, "Visions of Johanna", *Blonde on Blonde*, 1966

[2] Lillian Schwartz, "The Mona Lisa Identification", *The Visual Computer*, *vol. 4 p40-48*, 1988

[3] S. Russel, P. Norvig, "Artificial Intelligence: A Modern Approach" 2nd Edition, 2003

[4] S. Gerssen, "Masters Thesis: Bayesian Networks in Credit Rating", *p39-58*, 2004

[5] D. Heckerman, "A Tutorial on Learning with Bayesian Networks", 1996

[6] J. McClain Watson, "From interpretation to identification: a history of facial images in the sciences of emotion", *History of the Human Sciences*, *vol. 17*, *No. 1*, *p29-51,* 2004

[7] P. Ekman, W.V. Friessen, "Facial Action Coding System: Investigator's Guide", *Consulting Psychologists Press*, 1978

[8] M.A. Sayette, J.F. Cohn, J. M. Wertz, M.A. Perrot, D.J. Parrot, "A Psychometric Evaluation of the Facial Action Coding System for Assessing Spontaneous Expression", *Journal of nonverbal behavior*, *Vol. 25*, *p167-186*, 2001

[9] T. Kanade, J.F. Cohn, Y. Tian, "Comprehensive Database for Facial Expression Analysis", *The 4th IEEE International Conference on Automatic Face and Gesture Recognition, Proceedings*, *p46-53*, 2000

[10] Y. Yacoob, L. Davis, "Recognizing Facial Expressions by Spatio-Temporal Analysis", *12th International Conference on Pattern Recognition*, *p747-749*, 1994

[11] M. Pantic, L.J.M. Rothkrantz, "Automatic Analysis of Facial Expression: The State of the Art", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *Vol. 12*, *No. 12*, *p1424-1445*, 2000

[12] I. Cohen, N. Sebe, A. Garg, T S. Huang, "Emotion Recognition using a Cauchy Naïve Bayes Classifier", *International Conference on Pattern Recognition*, 2002.

[13] D. Datcu, L.M.J. Rothkrantz, "Machine Learning Techniques for Face Analysis", ICIS Project (2004)

[14] I. Cohen, N. Sebe, A. Garg, M.S. Lew, T S. Huang, "Facial Expression Recognition From Video Sequences", *International Conference on Multimedia and Expo (ICME)*, *Vol. 2*, *p121-124*, 2002

[15] I. Cohen, N. Sebe, Y. Sun, M.S. Lew, T S. Huang, "Evaluation of Expression Recognition Techniques", *International Conference on Image and Information Retrieval*, *p184-195*, 2003

[16] C. Liu, H Wechsler, "Probabilistic Reasoning Models for Face Recognition*", IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Proceedings*, *p827-832*, 1998

[17] L. Zalewski, S. Gong, "A Probabilistic Hierarchical Framework for Expression Classification", *Artificial Intelligence and the Simulation of Behaviour Symposium on Language, Speech and Gesture for Expressive Characters, Proceedings*, 2004

[18] D. Datcu, L.M.J. Rothkrantz, "Automatic recognition of facial expressions using Bayesian Belief Networks", *IEEE International Conference on Systems, Man and Cybernetics, Proceedings*, 2004

[19] N. Moënne-Loccoz, F. Brémond, M. Thonnat, "Recurrent Bayesian Network for the Recognition of Human Behaviors from Video", *The3rd International Conference on Computer Vision Systems, Proceedings*, 2004

[20] X. Li, Q. Ji, "Active Affective State Dectection and User Assistance With Dynamic Bayesian Networks", *IEEE International Conference on Systems, Man and Cybernetics*, *Part A*, *Vol. 35*, *Issue 1*, *p93-105*, 2005

[21] I. Cohen, N. Sebe, F.G. Cozman, M.C. Cirelo, T.S. Huang, "Learning Bayesian Network Classifiers for Facial Expression Recognition using both Labeled and Unlabeled Data", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Proceedings*, *Vol. 1*, *p595-601*, 2003

[22] J. Ghent, J. McDonald, "A Computational Model of Facial Expression", *NUIM-CS-TR-2004-0*, *technical report*, 2004

[23] J. Ghent, J. McDonald, "Facial Expression Synthesis using a Statistical Model of Appearance", *The 4th IASTED Conference on Visualization, Imaging and Image Processing*, 2004

[24] H. Wang, N. Ahuja, "Facial Expression Decomposition", *IEEE, The 9th International Conference on Computer Vision, Proceedings*, *p958-965*, 2003

[25] G. Guo, C.R. Dyer, "Learning From Examples in the Small Sample Case: Face Expression Recognition", *IEEE Transactions on Systems, Man and Cybernetics*, *Part B*, *Vol. 35*, *Issue 3*, *p477-488*, 2005

[26] SMILE and GeNIe, *Decision Systems Laboratory*, *University of Pittsburgh*, (www.sis.pit.edu/~genie), 1998