Graph-Structure Discovery in a Logical Probabilistic Model as a Form of Creativity

Greg Coppola
coppola.ai

January 6, 2025

Abstract

We have previously presented what we called a *logical graphical model*. In this work, we further clarify that an important aspect of the model is that it is *probabilistic*. Thus, we will now begin to refer to this as a "logical probablistic model". The logical probabilistic model is a kind of a Bayesian Network. One important problem that has always been a blocker for the practical use of Bayesian Networks is that the *structure* of a Bayesian Network must somehow be *specified*. This is a problem that Neural Networks do not have.

Contents

1 Introduction **3**

2 Maximum-Likelihood Estimation **4**
2.1 Maximum Likelihood Estimation 4
2.1.1 Principles of MLE 4
2.1.2 The Maximization Process 4
2.1.3 Properties of MLE 5
2.1.4 Applications of MLE 5
2.2 Maximizing the Likelihood of the Data 5
2.2.1 Objective of Maximization 6

2.2.2 The Likelihood Function 6
2.2.3 Use of Log-Likelihood 6
2.2.4 Optimization Techniques 6
2.2.5 Challenges in Maximization 7

3 Compressing the Data **7**
3.1 Theoretical Background 8
3.2 Entropy and Maximum Likelihood 8
3.2.1 Optimal Compression Algorithm 8
3.2.2 Maximum Likelihood and Compression Efficiency . . . 8
3.3 Practical Implications 9
3.4 Challenges and Future Directions 9
3.5 Predictions on Unseen Data 9
3.5.1 Generalization 9
3.5.2 From Fitting to Predicting 10
3.5.3 Predictive Performance 10
3.5.4 Techniques to Enhance Predictions 10
3.5.5 Evaluating Predictions 11

4 Hill-Climbing on an Objective Function **11**
4.1 Objective Functions 11
4.2 Optimization of An Objective Function 11
4.3 Expectation Maximization 12
4.4 The Crucial Role of the Objective Function 12

5 Principal Operations in a Bayesian Network **12**
5.1 Introduction 12
5.1.1 Definition 12
5.1.2 Calculating the Joint Probability Distribution 13
5.1.3 Steps to Compute the Joint Distribution 13
5.1.4 Example 13
5.2 Inference 14
5.2.1 Types of Inference 14
5.2.2 Methods for Inference 14
5.2.3 Applications of Inference 15
5.2.4 Challenges in Inference 15
5.3 Bayesian Parameter Learning 15
5.3.1 Bayes' Theorem 15

5.3.2 Choosing Priors 16
5.3.3 Computational Techniques 16
5.3.4 Applications and Advantages 16
5.4 Automatic Discovery of the Structure of a Bayesian Network . 17
5.4.1 The Challenge 17
5.4.2 Scoring Functions 17
5.4.3 Search Algorithms 18
5.4.4 Constraints and Priors 18
5.4.5 Challenges and Considerations 18
5.5 Model Structure in Neural Networks vs. Bayesian Networks . 19
5.5.1 Neural Networks: Fixed Structure Learning 19
5.5.2 Bayesian Networks: Structure and Parameter Learning 19
5.5.3 Comparative Analysis 20

6 Learning Structure with Only Observed Variables 21
6.1 The Case of Only Observed Variables 21
6.2 Review of Strategies in the Literature 21

7 Learning Structure with Hidden Variables 21
7.1 The Case of Hidden Variables 21
7.2 Review of Strategies in the Literature 21

8 Discussion 21
?

1 Introduction

We have previously presented what we called a *logical graphical model*. In this work, we further clarify that an important aspect of the model is that it is *probabilistic*. Thus, we will now begin to refer to this as a "logical probablistic model". The logical probabilistic model is a kind of a Bayesian Network. One important problem that has always been a blocker for the practical use of Bayesian Networks is that the *structure* of a Bayesian Network must somehow be *specified*. This is a problem that Neural Networks do not have.

This is a central point:

- **Mind-Blowing Insight**: The difference between neural networks and probabilistic graphical networks is that neural networks only require pa-

rameter training and inference, while probabilistic graphical models require inference, parameter learning and also something more difficult—the learning of the *structure* of the network.

2 Maximum-Likelihood Estimation

One of the central aspects of Ilya's talk is that he talks about the relationship between "prediction" and "compression" as a reason why a technique called "distribution matching" might explain the success of the "large language model".

Just in general, we might observe that there are deep connections between the following concepts.

2.1 Maximum Likelihood Estimation

Maximum Likelihood Estimation (MLE) is a statistical method used for estimating the parameters of a probability distribution by maximizing a likelihood function. The method is widely used to infer the parameters that best explain the observed data under a specified statistical model.

2.1.1 Principles of MLE

The likelihood function measures the probability of observing the given data as a function of the parameters of the model. In the case of MLE, the objective is to find the parameter values that maximize this likelihood function. Mathematically, the likelihood of a model parameter θ given observations X is expressed as:

$$L(\theta \mid X) = P(X \mid \theta)$$

Where $P(X \mid \theta)$ denotes the probability of observing X given the parameters θ.

2.1.2 The Maximization Process

To find the parameter values that maximize the likelihood function, one typically takes the logarithm of the likelihood, known as the log-likelihood, because it transforms the product of probabilities into a sum, simplifying the differentiation:

$$\log L(\theta \mid X) = \log P(X \mid \theta)$$

The parameters that maximize the log-likelihood are found by setting the derivative of the log-likelihood with respect to θ to zero and solving for θ. This process is known as finding the maximum likelihood estimators (MLEs) of the parameters.

2.1.3 Properties of MLE

Maximum likelihood estimators have several desirable statistical properties:

- **Consistency:** As the sample size increases, the MLE converges to the true parameter value.
- **Efficiency:** MLEs achieve the lowest possible variance among all unbiased estimators, especially as the sample size grows large, known as being asymptotically efficient.
- **Normality:** Under certain regularity conditions, the distribution of MLEs approaches a normal distribution as the sample size increases, which simplifies inference.

2.1.4 Applications of MLE

Maximum likelihood estimation is used in various fields such as economics, epidemiology, finance, and more. It is particularly prevalent in machine learning, both in supervised and unsupervised learning scenarios, where the likelihood can be used to train models like logistic regression, generalized linear models, and more complex structures where parameters are not directly observable.

In summary, maximum likelihood estimation is a cornerstone of statistical inference, providing a flexible and robust framework for parameter estimation across a wide range of models. Its widespread application in both theoretical and applied statistics underscores its fundamental role in the analysis of probabilistic models.

2.2 Maximizing the Likelihood of the Data

Maximizing the likelihood of the data is a fundamental goal in many statistical models and machine learning algorithms. This process involves adjusting the model parameters to make the observed data as probable as possible under the model.

2.2.1 Objective of Maximization

The primary objective of maximizing the likelihood is to find the parameter set θ that makes the observed data most probable. This approach is rooted in the principle that the best model for the data is the one under which the data is most likely to occur.

2.2.2 The Likelihood Function

The likelihood function $L(\theta \mid X)$, representing the probability of observing the data X given parameters θ, serves as the cornerstone for this maximization process. The function is often expressed as:

$$L(\theta \mid X) = \prod_{i=1}^{n} f(x_i \mid \theta)$$

where $f(x_i \mid \theta)$ is the probability density or mass function, and n is the number of data points. This product can be cumbersome to work with due to numerical underflow or computational inefficiencies, particularly with large datasets.

2.2.3 Use of Log-Likelihood

To address these challenges, the log-likelihood function is used:

$$\log L(\theta \mid X) = \sum_{i=1}^{n} \log f(x_i \mid \theta)$$

Maximizing this sum is mathematically equivalent to maximizing the product of the original likelihood but is computationally more stable and easier to handle, especially when differentiating with respect to θ.

2.2.4 Optimization Techniques

Various optimization techniques can be employed to maximize the log-likelihood, depending on the complexity of the function and the model:

- **Gradient Descent:** Used for models where the gradient of the log-likelihood can be computed, allowing iterative improvement of the parameter estimates.

- **Newton-Raphson:** A more sophisticated method that uses both the first and second derivatives of the log-likelihood to update the parameters, which can converge faster than gradient descent.
- **Expectation-Maximization (EM):** For models with latent variables, the EM algorithm can be used to iteratively estimate the hidden variables and maximize the likelihood.

2.2.5 Challenges in Maximization

While the maximization of the likelihood is conceptually straightforward, several challenges can arise:

- **Multiple Local Maxima:** The likelihood function may have multiple peaks, especially in complex models, making it difficult to find the global maximum.
- **Overfitting:** Maximizing the likelihood without regularization can lead to overfitting, where the model describes random error or noise instead of the underlying relationship.
- **Computational Complexity:** For large datasets or complex models, the computation of the likelihood and its derivatives can be computationally intensive.

In summary, maximizing the likelihood of the data is a powerful strategy in statistical modeling and machine learning. It involves sophisticated mathematical and computational techniques to refine model parameters, ensuring that the model accurately captures the patterns in the observed data.

3 Compressing the Data

The concept of compressing data involves reducing the amount of data needed to represent a given set of information accurately. An optimal compression algorithm minimizes the length of the encoded data while preserving the information content. The relationship between such algorithms and maximum likelihood estimates is profound, as both seek to optimize a measure of how well the model (or compression scheme) represents the observed data.

3.1 Theoretical Background

At the heart of data compression lies the principle of minimizing redundancy, which corresponds to maximizing the efficiency of the data representation. Information theory, particularly the work of Claude Shannon, provides a foundational framework linking probabilistic models of data with the limits of data compression.

3.2 Entropy and Maximum Likelihood

Entropy, a measure of uncertainty or randomness in data, is a key concept in information theory and provides a lower bound on the average length of the shortest possible lossless encoding of the data. The connection to maximum likelihood estimation arises when considering that the best compression of a dataset is achieved by a model that most accurately describes its underlying probability distribution.

3.2.1 Optimal Compression Algorithm

An optimal compression algorithm can be conceptualized as one that encodes data by precisely matching the data's inherent probability distribution. From a practical standpoint, this means:

- **Model Fitting:** Using maximum likelihood estimation to fit a probabilistic model to the data, thereby determining the parameters that best describe its distribution.
- **Entropy Encoding:** Applying entropy-based coding techniques (such as Huffman coding or arithmetic coding) that use the fitted model to assign shorter codes to more probable data points and longer codes to less probable ones.

3.2.2 Maximum Likelihood and Compression Efficiency

The maximum likelihood estimate of a model's parameters ensures that the estimated probability distribution of the model is as close as possible to the true distribution of the underlying data. Therefore, coding schemes based on these estimates are more efficient, reducing the expected code length toward the theoretical minimum described by the entropy of the model.

3.3 Practical Implications

In practice, the interplay between maximum likelihood estimation and data compression unfolds in various ways:

- **Data Compression Software:** Tools that compress text, audio, and video data often employ models of the data (e.g., Markov models for text or Gaussian models for audio) that are parameterized via methods akin to maximum likelihood.
- **Noise Reduction:** In signal processing, the optimal compression of signals (removing noise) is frequently achieved by modeling the signals with parameters estimated via maximum likelihood, enhancing both the compression and clarity of the signal.

3.4 Challenges and Future Directions

While maximum likelihood provides a robust framework for modeling and compression, it is not without challenges. These include the potential for overfitting in small data samples and computational burdens in parameter estimation for large datasets. Future directions may involve more sophisticated models that balance model complexity with compression efficiency, possibly integrating machine learning techniques to adaptively fit models in dynamic environments.

In conclusion, understanding and leveraging the relationship between maximum likelihood estimation and optimal data compression can lead to significant improvements in how data is stored, transmitted, and analyzed, making efficient use of resources while maintaining data integrity.

3.5 Predictions on Unseen Data

Predicting outcomes on unseen data is a central goal of many statistical models and machine learning algorithms. After developing models based on historical data, the true test of their utility comes when they are used to make predictions or decisions about future or unknown events.

3.5.1 Generalization

Generalization refers to a model's ability to perform well on new, unseen data, not just on the data on which it was trained. This ability is crucial for

the practical application of predictive models in real-world scenarios.

3.5.2 From Fitting to Predicting

While models are often fitted to data using techniques such as maximum likelihood estimation, the ultimate aim is to use these models to make accurate predictions. Here's how the process typically unfolds:

- **Model Training:** A model is trained on a dataset, using methods like maximum likelihood estimation to determine the parameters that best fit the training data.
- **Model Validation:** The model is then tested against a separate validation dataset to assess its predictive accuracy and to tune model hyperparameters, reducing the risk of overfitting.
- **Prediction:** Finally, the model is used to make predictions on new, unseen data, applying the learned parameters to infer or forecast outcomes.

3.5.3 Predictive Performance

Several factors influence the predictive performance of a model on unseen data:

- **Model Complexity:** While more complex models can fit the training data better, they may also overfit, capturing noise rather than the underlying data pattern, and thus perform poorly on unseen data.
- **Size of Training Data:** Larger datasets generally provide more information and lead to better generalization.
- **Noise and Variability:** The amount of noise in the training data can affect the model's ability to generalize, as high noise levels can lead to models that are less robust.

3.5.4 Techniques to Enhance Predictions

To improve the model's performance on unseen data, several techniques are commonly employed:

- **Cross-Validation:** This technique involves repeatedly splitting the data into training and test sets, ensuring that the model performs well across different subsets of the data.
- **Regularization:** Techniques like L1 or L2 regularization are used to prevent overfitting by adding a penalty on the size of the coefficients.
- **Ensemble Methods:** Combining multiple models to make predictions can reduce variance and improve prediction accuracy on new data.

3.5.5 Evaluating Predictions

Ultimately, the effectiveness of a model on unseen data is assessed using specific performance metrics, such as accuracy, precision, recall, F1 score for classification tasks, or mean squared error and R-squared for regression tasks.

In summary, the ability to make reliable predictions on unseen data is what distinguishes a theoretically good model from a practically effective one. Techniques that enhance generalization are vital to developing models that are not only statistically sound but also robust and applicable in real-world scenarios.

4 Hill-Climbing on an Objective Function

4.1 Objective Functions

Objective functions are mathematical models that measure the performance or output of a system based on a set of input variables. In optimization, the objective function is what we aim to maximize or minimize. For instance, in machine learning, an objective function can be used to evaluate the accuracy of a model or the error between the predicted and actual values.

4.2 Optimization of An Objective Function

Optimization involves finding the best solution from a set of feasible solutions. Hill-climbing is a simple heuristic used in optimization that iteratively moves towards a higher value of the objective function. It starts from an arbitrary point and makes local changes that increase the objective function's value. This technique is particularly useful when the search space is large, and finding the absolute maximum (or minimum) is computationally infeasible.

4.3 Expectation Maximization

Expectation Maximization (EM) is a statistical technique for finding maximum likelihood estimates in models with latent variables. It alternates between performing an expectation (E) step, which computes an expectation of the likelihood by including the latent variables as if they were observed, and a maximization (M) step, which computes the maximum likelihood estimates with the expected values obtained in the E step. This process iterates until convergence. EM can be thought of as hill-climbing because each iteration aims to increase the likelihood function.

4.4 The Crucial Role of the Objective Function

The objective function is crucial in any optimization problem because it defines the goal of the optimization. In hill-climbing, the choice of objective function significantly impacts the effectiveness and efficiency of the search. A poorly chosen objective function might lead the algorithm to converge to local optima rather than the global optimum, especially in complex landscapes. Therefore, the design of the objective function must align closely with the ultimate goals of the optimization process.

5 Principal Operations in a Bayesian Network

5.1 Introduction

5.1.1 Definition

A Bayesian network for a set of variables $X_1, X_2, \ldots, X_n$ consists of:

- A set of nodes, each representing a variable X_i.
- A set of directed edges that connect pairs of nodes, representing conditional dependencies.

The key property of Bayesian networks is that each variable is conditionally independent of its non-descendants given its parent nodes in the graph.

5.1.2 Calculating the Joint Probability Distribution

The joint probability distribution of the variables in a Bayesian network is given by the product of conditional probabilities for each variable, conditioned on its parents in the network. Mathematically, it is expressed as:

$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i \mid \text{Parents}(X_i))$$

where $\text{Parents}(X_i)$ denotes the set of parent nodes of X_i in the network.

5.1.3 Steps to Compute the Joint Distribution

1. **Identify the Parents**: For each variable X_i, identify its parents in the Bayesian network. If X_i has no parents, the conditional probability reduces to the marginal probability of X_i.

2. **Determine Conditional Probabilities**: Utilize the conditional probability tables (CPTs) provided with the Bayesian network. These tables specify the probability of each variable for every combination of values of its parents.

3. **Compute the Product**: For a particular instantiation of all variables $x_1, x_2, \ldots, x_n$, compute the product of the conditional probabilities from the CPTs. This product gives the probability of that particular instantiation.

5.1.4 Example

Consider a simple Bayesian network with three variables: A, B, and C, where A causes B and B causes C (i.e., $A \rightarrow B \rightarrow C$).

The joint distribution is calculated as:

$$P(A, B, C) = P(C \mid B) \cdot P(B \mid A) \cdot P(A)$$

If you know that:

- $P(A = a_1) = 0.3$,
- $P(B = b_1 \mid A = a_1) = 0.5$,

- $P(C = c_1 \mid B = b_1) = 0.4$,

Then:

$$P(A = a_1, B = b_1, C = c_1) = 0.4 \cdot 0.5 \cdot 0.3 = 0.06$$

5.2 Inference

Inference in Bayesian networks is the process of computing the posterior distribution of certain variables given evidence about other variables. This allows for probabilistic reasoning about the variables based on observed data. There are primarily two types of inference tasks: marginalization and conditional probability queries.

5.2.1 Types of Inference

Marginalization Marginalization involves computing the probability distribution over a subset of variables, integrating out or summing over the other variables. This is useful when the interest lies in the overall behavior of a few variables without considering the entire network.

Conditional Probability Conditional probability queries, on the other hand, involve calculating the probabilities of certain outcomes given the known values of other variables. This is often expressed as $P(X \mid e)$, where X is the query variable and e represents evidence, i.e., known values of other variables.

5.2.2 Methods for Inference

Exact Inference Exact inference techniques, such as the junction tree algorithm, involve converting the Bayesian network into a tree structure where the inference can be performed efficiently. Although powerful, exact inference can be computationally prohibitive in large networks.

Approximate Inference Approximate inference methods are often used when exact inference is not feasible. Techniques such as Monte Carlo simulations, including Markov Chain Monte Carlo (MCMC) and Gibbs sampling, provide ways to estimate the distribution with a degree of uncertainty. Another popular method is loopy belief propagation, which uses message passing in networks with cycles, albeit without guarantees of convergence.

5.2.3 Applications of Inference

Inference in Bayesian networks is widely used in various fields such as genetics, where it helps in understanding the genetic disposition to diseases, in diagnostics, where it aids in deriving patient-specific probabilities of having certain conditions, and in machine learning, particularly in areas involving uncertainty and prediction.

5.2.4 Challenges in Inference

Despite its utility, Bayesian inference faces challenges, especially regarding scalability and computational efficiency. The complexity of the network and the interdependencies between variables can make inference computationally intensive, which requires innovative solutions for practical applications.

In summary, inference is a fundamental aspect of Bayesian networks, enabling the extraction of meaningful insights from complex probabilistic models. As computational resources grow and techniques advance, the scope and accuracy of Bayesian inference continue to expand, making it an indispensable tool in the realm of probabilistic modeling and decision-making.

5.3 Bayesian Parameter Learning

Bayesian parameter learning refers to the process of updating the probabilistic models' parameters based on observed data, utilizing Bayes' theorem. This approach contrasts with frequentist methods by incorporating prior knowledge or beliefs about the parameters before observing the data. The goal is to compute the posterior distribution of the parameters, which combines prior beliefs with the likelihood of observing the data given those parameters.

5.3.1 Bayes' Theorem

The foundation of Bayesian learning is Bayes' theorem, which in the context of parameter learning, is expressed as:

$$P(\theta \mid D) = \frac{P(D \mid \theta)P(\theta)}{P(D)}$$

where θ represents the parameters of the model, D is the observed data, $P(\theta)$ is the prior distribution of the parameters, $P(D \mid \theta)$ is the likelihood of the

data given the parameters, and $P(D)$ is the evidence or the probability of the data under all possible parameter values.

5.3.2 Choosing Priors

The choice of prior $P(\theta)$ is critical in Bayesian learning. Priors can be:

- **Informative priors**: These contain specific information about what values the parameters might take based on previous studies or expert knowledge.
- **Non-informative priors**: These are designed to have minimal impact on the posterior distribution, often used when little prior knowledge is available.
- **Conjugate priors**: These are selected because they lead to posterior distributions that are of the same family as the prior, simplifying the computation.

5.3.3 Computational Techniques

Updating beliefs in light of new data typically requires integration over many possible parameter values, which can be computationally challenging. Common techniques used in Bayesian parameter learning include:

- **Markov Chain Monte Carlo (MCMC)**: This is a class of algorithms that samples from the posterior distribution when direct sampling is complex.
- **Variational Inference**: This method approximates the posterior by a simpler distribution by solving an optimization problem.

5.3.4 Applications and Advantages

Bayesian parameter learning is widely used in fields where the stakes of decision-making are high, such as in clinical trials, finance, and policy modeling. The ability to update the model as new data arrives allows for more flexible and adaptive decision-making. Moreover, the incorporation of prior knowledge helps in situations where data might be scarce or expensive to obtain.

In summary, Bayesian parameter learning offers a robust framework for understanding uncertainty in parameter estimates and making informed decisions based on both prior knowledge and new data. The flexibility in choosing prior models and the richness of the posterior analysis are key advantages of this approach over more traditional methods.

5.4 Automatic Discovery of the Structure of a Bayesian Network

Automatic discovery of the structure of a Bayesian network involves determining the most probable graphical model that represents the dependencies among a set of variables based on observed data. This task is crucial in many applications where the underlying dependency structure is not known a priori and must be inferred from data.

5.4.1 The Challenge

The primary challenge in learning the structure of a Bayesian network from data is that the number of possible structures grows super-exponentially with the number of variables. Hence, finding the optimal network structure is computationally intensive and often NP-hard.

5.4.2 Scoring Functions

To evaluate how well a particular structure fits the data, various scoring functions are used. These include:

- **Bayesian Information Criterion (BIC)**: Balances model complexity against the model's goodness of fit.
- **Akaike Information Criterion (AIC)**: Similar to BIC but with a different penalty for the number of parameters.
- **Bayesian Dirichlet equivalent (BDeu)**: A Bayesian score that incorporates prior beliefs about the network structure.

The goal is to maximize these scores over all possible structures, which indirectly maximizes the likelihood of the data given the structures.

5.4.3 Search Algorithms

Due to the vast number of potential network structures, heuristic search methods are employed to explore the space:

- **Greedy Search**: Starts with an empty graph or a random structure and iteratively adds, removes, or reverses edges to improve the scoring function.
- **Hill Climbing**: A variant of greedy search that makes local changes to the structure and accepts changes that improve the scoring.
- **Tabu Search**: Enhances greedy algorithms by avoiding cycles in the search space, using a tabu list that prevents revisiting recent configurations.
- **Genetic Algorithms**: Use principles of evolution to explore various structures through operations like mutation and crossover based on their fitness.

5.4.4 Constraints and Priors

Incorporating domain-specific knowledge as constraints or priors can significantly reduce the search space and improve the learning efficiency. Constraints can specify which variables are more likely to be parents of others, whereas priors can influence the initial probabilities of certain network configurations.

5.4.5 Challenges and Considerations

The automatic discovery process must balance accuracy with computational feasibility. While more complex models may better fit the data, they are also more prone to overfitting and can be computationally prohibitive. Moreover, the choice of scoring function and search algorithm can greatly affect the quality of the discovered structure.

In summary, the automatic discovery of the structure of Bayesian networks is a complex but valuable process that helps in understanding the underlying probabilistic models in various domains. Advances in computational techniques and algorithms continue to improve the efficacy and efficiency of this process, making it a vital tool in data-driven decision-making.

5.5 Model Structure in Neural Networks vs. Bayesian Networks

Understanding the structural learning differences between neural networks and Bayesian networks is crucial for selecting the appropriate modeling approach based on the nature of the problem and the data available.

5.5.1 Neural Networks: Fixed Structure Learning

Neural networks are characterized by a predefined architecture, which includes the number of layers, the number of neurons in each layer, and the connection patterns between the neurons (e.g., fully connected, convolutional). The learning process in neural networks involves adjusting the weights of the connections through training, typically using backpropagation and gradient descent methods. This fixed structure implies that the model's capacity and its ability to capture complex patterns are determined before training begins.

Key Characteristics:

- **Architecture Selection:** The structure or topology of a neural network is usually decided based on the specific application and is often guided by empirical performance metrics, prior knowledge, or experimentation.
- **Parameter Optimization:** Once the architecture is set, learning focuses solely on optimizing the weights of the connections to minimize a loss function, which measures the difference between the predicted and actual outputs.

5.5.2 Bayesian Networks: Structure and Parameter Learning

Unlike neural networks, Bayesian networks require learning both the parameters and the structure from data unless the structure is predefined based on domain knowledge. The structure of a Bayesian network represents conditional dependencies between variables, which needs to be determined from the data if not known a priori. This dual learning requirement makes Bayesian networks particularly suited for exploratory analysis where the relationships among variables are not fully understood.

Key Challenges:

- **Structural Discovery:** The process of learning the structure involves determining which nodes (variables) are directly connected and in which direction. This can be computationally intensive, as it involves evaluating the fit of numerous possible graphs.
- **Parameter Estimation:** Once the structure is determined, the next step is to estimate the conditional probability tables (CPTs) for each node given its parents, which quantify the relationships encoded by the structure.

5.5.3 Comparative Analysis

The necessity to determine structure in Bayesian networks adds a layer of complexity not present in neural networks, where the focus is primarily on parameter optimization. This fundamental difference stems from the purposes these models serve:

- **Neural Networks:** Optimized for prediction and performance, particularly in scenarios where large volumes of data are available.
- **Bayesian Networks:** Best suited for inference and understanding the underlying probabilistic relationships between variables, especially useful in domains where interpretability and the quality of uncertainty estimates are important.

In summary, the approach to model structure significantly impacts the application and effectiveness of neural networks and Bayesian networks. While neural networks excel in handling complex, high-dimensional data for predictive tasks, Bayesian networks offer advantages in reasoning about the probabilistic interactions and causal relationships among variables, provided their structure can be accurately learned from the data.

6 Learning Structure with Only Observed Variables

6.1 The Case of Only Observed Variables

6.2 Review of Strategies in the Literature

7 Learning Structure with Hidden Variables

7.1 The Case of Hidden Variables

7.2 Review of Strategies in the Literature

8 Discussion

We suggest that one aspect of "creativity" is actually the *learning* of structure in a belief network.

References