Bayesian Networks Are Not the Solution to Opening Machine Learning's Black Box

Andrew Parisi and Casey Hart

April 2020

Abstract

Bayesian networks are causal/probabilistic models that are significantly more transparent than other "black box" ML solutions. However, these graphs are also extremely limited: they can only express a single type of relation between nodes, and the nodes themselves are inflexible. CycL, on the other hand, permits richer comparison between entities. Cyc is also designed to scale and generate causal (and other kinds of) reasoning in the face of uncertainty. Bayesian networks are fantastic tools for limited models with certain assumptions, but one should not mistake these networks as a transparent solution for ML's black box problem.

1 Bayesian Networks: Brief Overview and Advantages

Bayesian networks are node and link graphs; they encode causal/correlative data in an acyclic graph where each node in that graph corresponds to an event or true statement, and each link indicates the strength or probability that one can/should/will "follow" that link while reasoning.

Given (i) some values at the root or roots of the graph and (ii) a method of computing the probability of each node given the probabilities of its parent nodes, it is then possible to follow that method iteratively and compute the probability of any node in the graph.

The resulting graph can be inspected and processed so as to provide a kind of explanation for each conclusion that it draws. Suppose, for example, that there is a Bayesian graph having a link that connects a node labelled "It is raining" to a node labelled "The ground is wet", and another link from that node to one labelled "The ground is slippery". Then when that graph is used to infer that the ground is slippery from the claim that it is raining, one can "read off" the step by step path that leads from the givens (the roots – in this case the node labelled "It is raining") to the final conclusion and – to the extent that the inference method (ii, above) is natural and obvious and valid – that path will be seen as natural and clear and valid to a human examining the path.

The inference in the last paragraph could be generated using a very small amount of data. It could even be the case that experts were used to enter the nodes in the graph and their connections. This is in contrast to machine learning approaches that offer no explanation for the conclusions they generate and make it particularly difficult for experts to enter any sort of data to manipulate the inferential path that has been taken. In this sense, Bayesian networks are clearly a step on the right path towards artificial intelligence that would be fully transparent and editable while also only requiring a relatively small amount of data (or even no data) to work.

2 A Deeper Analysis of Bayesian Networks

To be a little more precise than we were above, a Bayesian network is a graphical representation of probability distributions. There are three components:

Nodes: Each node in a Bayesian network graph represents a proposition that fixes the value of some variable. For example: (Smoker = True) or (Age = 32) or (Distribution of balls in urn = 3 blue, 2 red, 5 yellow).

Directed Edges: Nodes are connected to other nodes by arrows; these models are often called DAGs for "Directed Acyclic Graph." Arrows typically represent causality. So, if A is a (partial) cause of B, one can make nodes for A and B and connect them: A B.

Probabilities: These causal or correlation edges need not be guaranteed causes. Smoking is a cause of cancer, but not all smokers get cancer. Bayesian networks allow us to insert the conditional probability, such as P(Cancer — Smoking) = .7, "the probability of cancer on the condition that one smokes is 70%."

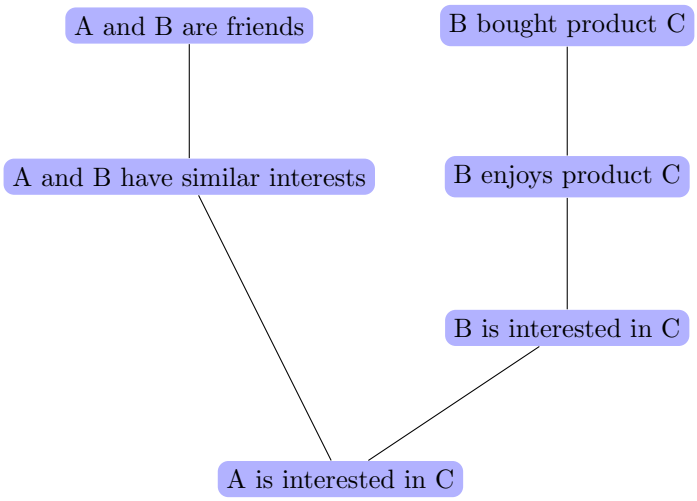
What makes these graphs *Bayesian* is their application of Bayes' Theorem, among other probability equations. Bayes' Theorem states that P(A—B) = P(A)*P(B—A)/P(B), as long as P(B) is nonzero. Once such a graph is completed, updating the status of a single node will ripple forward and backward through the arrows and generate posterior probabilities for each node.

Links between graph nodes generally represent one sort of inferential link: causes, or caused by, or co-occurs with, or correlates with, etc. Or else the graph takes the risk of mixing together several different meanings for a link, which makes the conclusion difficult to understand and difficult to trust. A combinatorial explosion happens if one tries to scale up these types of graphs to include large numbers of nodes that have a wide variety of possible values. Thus, links in a Bayesian network are often interpreted as causal. To a hammer, everything looks like a nail; to a Bayesian network, everything looks like a causal structure. For sprinklers and wet streets, this is entirely appropriate. But a link between "a person has a male sibling" to "a person has a brother" is definitional, not causal. Moreover, a Bayesian network link often just means "there is some

third factor at work here which happens to often cause A and B" –consider the link between "a woman is in a hospital" and "a woman has a newborn infant." If you discover that a woman is in the hospital, and nothing else, then your confidence that she has a newborn infant should increase. But it would be a mistake to think that the woman's being in the hospital might cause her to have a newborn baby, nor should one infer that possessing the baby caused her to be in the hospital. Instead, we know that there is a common cause for both: giving birth.

Since not every bit of reasoning involves causation, one might try interpreting some arrows as meaning something other than 'causes'. In the case of the link between brother and male sibling (everyone who has a male sibling has a brother, and so the P(brother—male sibling) = 1), the link is a semantic or definitional one. The problem is that so far as the reasoning method (ii, above) is concerned, there is only ever one type of link in a given Bayesian network graph.

Consider the following example: A company wants to model what users they expect will enjoy which products they sell. One explanation for why a user may like a product is that they have a friend who has also bought that product. In this case, we can set up a Bayesian network with a link from a person having a friend who owns a product to a person being likely to enjoy that product. Now someone may want to know why that link is in the graph. In this case, it is ambiguous, the link could be causal: friends tend to tell their friends what they enjoy. The link might also be due to a less causal path, friends tend to have similar interests, and people are interested in things they enjoy. Since there are no graphs with different types of links, the simple graph we just built is impoverished in the explanations it can generate. We could distinguish what sort of link we intended to set up by using the more complicated graph with six nodes that looks something like this:



This graph may offer some explanation of the relationship between A and B being friends, B buying product C, and A being interested in C, but there is no uniform interpretation of what a link means. For the top right link from 'B bought product C' to 'B enjoys product C' there is probably a causal interpretation of that link from bottom to top. B enjoying a product may cause them to buy it. A causal explanation will not work for either of the bottom two links though. A being interested in C and having similar interests to C does not cause B to be interested in C. Instead those links from bottom to top encode something about what the words 'similar interest' mean. Those links are better characterized as semantic links between nodes. A good and clear explanation of why A would be interested in product C should include all of these links along with information about what kind of link it is. (Spoiler: Cyc is able to express as many types of relations as anyone would like, and in fact already has tens of thousands of types of relations with fully fleshed-out semantics.)

Bayesian networks perform well on small graphs where nodes have a relatively small number of possible values. Importantly, for each link in a graph, the probability of that link given its parent links needs to be calculated. The number of conditional probabilities that need to be counted is exponential given the number of parents that a node has. In the above graph, all of the nodes could take only two values. The node 'A is interested in C' has 5 parents. Therefore, in order to calculate the conditional probability table for that node, we would need to make 25 calculations. That's for a graph with only six nodes and only binary values to be assigned to that node. Considering a more scaled-up situation where a company might want to track its customers' purchase history, its products, and its customers' friends, the values for these nodes would be much greater than 2. Suppose that the company has 200 customers and 300 products, and each customer could be friends with 100 people. Suppose we care about a node with only two parents that takes friends as values but relies on a node that takes customers as a parent, which takes products as its parent: in this case, the number of entries in that node's conditional probability table would be 100200*300. That number is astronomical, and we have only considered three nodes with a very small number of values for each node. A company that only had 200 customers does not need Bayesian networks to figure out what its customers enjoy, and already they would be stalling out. Although there are ways to prevent this exponential growth, none of them would apply to the graph that concludes with 'A is interested in C'. And if we want those nodes to take more values than True and False, then things spiral quickly.

Bayesian networks are more transparent than a purely statistical ML black box – e.g., a multi-layer neural network. Neural nets use complex math to churn out predictive correlations, but they require big data, and their methods are opaque. In the end, if a neural net gives some output like "buy this stock", the justification will just be: "because complex mathematics says so". This is not trustable, and it certainly is not compelling to the SEC. On the other hand, a Bayesian network will justify a conclusion like "stock X will increase with .9 certainty" with something like: "we have built a causal model of the world, which has that value as the output". This is better, since you can at least

partially examine this causal model: we can look at the nodes and see tables that express the conditional probabilities that tell us how tightly (causally) connected these variables are. But there are some noteworthy absences:

2.1 We will not always know how the structure was determined. If it was statistically derived, the structure itself may be as much of a black box as a neural net.

2.2 We will not always know how the conditional probabilities were determined. Again, if ML was used to generate these conditional probabilities, it may be opaque to users.

2.3 These explanations still bottom out in being able to understand a graphical model.

2.4 All of this leads to affirmative conclusions; plausible arguments against Q being true are not generally computable, and hence not capable of being "read off" and articulated.

A satisfying explanation should make clear the reasons that support a conclusion or argue against it. In summary, then, Bayesian networks are powerful tools for modeling clearly defined causal structures, but they face issues in terms of scalability, transparency, and pro/con argumentation.

3 Cyc

In order for AI to really work and be useful it needs to be expressive, have auditable and clear explanations of its conclusions, and be able to scale well beyond small problem-spaces. It needs to generate answers to questions when there is a lot of data available and offer clear reasons for – and perhaps against – the answers that it is giving. When it makes a mistake, users of the AI need to be able to examine exactly why it made a mistake and then correct that mistake. Machine learning is inadequate for the latter reasons, and Bayesian networks are inadequate for the former ones.

Cyc produces a series of reasons for every answer it generates. Each answer is readily accessible to humans and can be stored for later reuse. The rules that Cyc uses are completely public; there is nothing hidden. Furthermore, its rules apply across a wide variety of domains and for a wide variety of reasons. Cyc's rules can encode mathematical knowledge, causal knowledge, semantic knowledge, knowledge about gardening and royalty – any kind of knowledge you can imagine. Cyc's expressiveness when it comes to the reasons it uses is as expressive as any natural language. For example, if you ask Cyc "Do fatigued lifeguards have trouble doing their job?", Cyc answers that this is true. But it also provides an explanation for why it's true. Cyc offers the following two facts in support:

3.1 Lifeguards have to be able to perform athletic activities for their jobs.

3.2 Fatigued people have trouble performing athletic activities.

If Cyc comes to any conclusions that look false, it can offer arguments like the above one for why it believes what it does. This makes it easy to understand why Cyc says what it says and to see exactly where it has gone wrong.

Note that the first reason in our lifeguard argument is not causal. It's something about what lifeguards need to be able to do. Cyc can offer a justification of that belief accordingly. Cyc knows that lifeguards have to be able to perform athletic activities because it also knows:

3.1a. Lifeguards have to be able to swim for their jobs.

3.1b. Swimming is an athletic activity.

This explanation has nothing to do with causation, and Cyc is clear about that.

All of Cyc's knowledge is fully explicit, declarative, and reusable. E.g., if you ask Cyc "Do you have to be able to move in order to swim?", it will answer "yes" because it knows that swimming is an athletic activity and in order for an agent to perform athletic activities they have to be able to move. This means that instead of having to write a new graph and calculate new conditional probabilities for something that a Bayesian network has never encountered, Cyc can draw on its vast knowledge base to answer a question.

Cyc's expansive knowledge base is proof that it is scalable. It knows 25,000,000 principles and rules of thumb that it can leverage at any time to generate answers to questions like the one above. Even the number of one-step inferences Cyc can draw is in the trillions, and it is not uncommon for Cyc to produce explanation graphs that have a small number of thousands of inference steps in them![1]

Cyc uses several types of meta-knowledge to enable it to quickly find the assertions and rules from its knowledge base that are relevant to the current problem. It also uses meta-knowledge to plan out how it should try to use those assertions and rules to find an answer. Given an answer, it can look back at the successful path to that answer in order to provide a step by step justification for that answer.

Any application that has access to Cyc can make use of the vast knowledge that Cyc has. That includes common sense knowledge (such as time and space and causality), intermediate level theories (of weather, traffic, emotions, etc.), and – when applicable – domain-specific knowledge. This domain knowledge can include non-proprietary assertions and rules added during the course of building the hundreds of previous Cyc applications – e.g., Ohm's Law, typical sizes of aortic tears, and the ICD-10 code for diabetes.

Moreover, Cyc can interface with external sources and in effect do a sort of virtual data integration or data-laking: it can reason as though the knowledge

[1] Cyc is optimized to quickly identify and use just the 0.001% of its knowledge – just the knowledge that's appropriate – when answering a specific question, but 0.001% of 25 million is 2,500, which is why a typical Cyc application query involves a couple thousand inference steps.

in those disparate knowledge sources were part of the Cyc knowledge base. Cyc accomplishes this by running out to access those databases and web services just as a human performing that task would. To make this happen, we translate the meanings of the external content into existing Cyc vocabulary. Because CycL is so expressive, it has the power to represent any information that might be in the external source.[2]

This means that Cyc can – and typically does – operate not by itself but synergistically. We partner with machine learning systems, Bayesian networks, and ontology-based solutions that use less expressive representations such as knowledge graphs and triple-stores. This allows us to maintain the expressivity of Cyc without losing the speed or advantages of these other formats in their respective niches.

4 Conclusion

Bayesian networks can be an asset in your AI toolkit, but they do have some limitations. On the one hand, they can provide powerful probabilistic inferences, revealing or unpacking causal structures to generate posterior probabilities for related events. On the other hand, these models are inflexible, not scalable, and relatively opaque. They can only express things in triples, being a node and link structure. Moreover, every one of these links must have the same meaning, and that meaning is usually restricted to 'causes' or 'raises the probability of'; more nuance is left to other more expressive languages. On the scaling front, the number of computations required even for very small models is quite large. At enterprise scale, Bayesian models of this sort for inference are untenable, even when given vast resources. Lastly, the transparency will boil down to a series of calculations carrying out Bayes' theorem. To be sure, this is more meaningful than a deep learning black box, but it falls far short of a reasoned argument.

As another tool in your AI toolkit, Cyc can help overcome some of the limitations of Bayesian networks, which are useful for probabilistic modelling but do not themselves constitute an overall AI solution. Cyc provides transparent logical reasoning across domains. To facilitate this reasoning, Cyc has an underlying representation language as expressive as any natural language (such as English): you are not restricted to only causal relationships. And with over 35 years of experience developing a variety of applications that leverage the same (ever-growing) knowledge base, Cycorp has a demonstrably scalable solution.

[2]For the same reason, Cyc has never failed to be able to fully capture a definition, equation, rule, or rule of thumb that a domain expert was able to articulate and communicate in English.