Bayesian nets
Meeting notes 10/19/2022

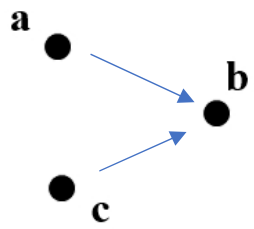
Amber, Dennis, Sophia and Patrick were physically present. Patrick will try to bring Zhongming up to speed by Zoom.

I. We started by talking about what Dennis and Amber had found in looking at the Bayesian programs. Consensus seemed to be in favor of Pyagrum—or at least using portions of Pyagrum within other programming if necessary.

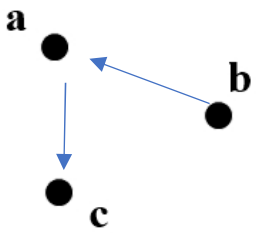
Dennis likes working in Spyder, so we'll go with that.

II. We then started working toward what we want in programming, with a miniature 'proof of concept' form of what we're after.

The components, as before: a 'world' that we construct. To keep it simple, just 3 nodes:



and an epistemic network intended to represent that world:



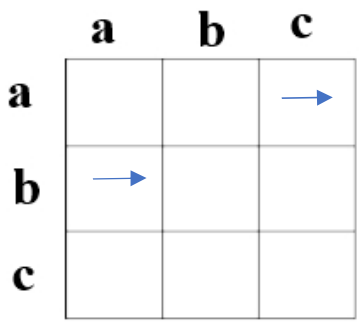
Clearly that representation might be more or less correct. In our model, the 'world' stays the same.

A major component of the model is that we want the representation to change, with changes tested against input from the world. The idea is that our representations can change in ways that make them more accurate.

We therefore need to model (a) the world, (b) representations, (c) how far off a representation is, judged in terms of (d) input from the world, and (e) a way of changing representations.

III. Here is the proposal for changing the network on the epistemic or representational side. It's a genetic algorithm, though here applied to networks in a way I haven't seen before.

Our representation, like that in the picture above, will correspond to an 'adjacency matrix.' This puts each of our nodes vertically on the left side, and again horizontally across the top, and puts an arrow (or an x, or a 1) in a box if there is a directed arrow from the left side node to the top node. Here's the adjacency matrix for our representation network above:



As a next step, we could represent the same information regarding our network by 'coding this up' with a 1 for an occupied square and a 0 otherwise. Reading left to right and up to down like text in a book, we can code our network as:

0 0 1 1 0 0 0 0 0 No arrow in the first two squares, then an arrow then an arrow, the no more arrows….

If our networks don't allow a directed link from a node directly to itself, nothing will ever occupy that diagonal a to a, b to b, etc. Leaving those squares out, we could code out network more economically as:

0 1 1 0 0 0

Here's why this is useful. We can make changes in our network by mutating this code—flipping one of these digits at random. If we changed it to

0 1 1 1 0 0,

for example, we would have added a directed link from b to c. Progressive mutations allow a changing network, which could be gauged for accuracy at each stage, keeping the most accurate networks yet achieved and further mutating them in exploration from there.

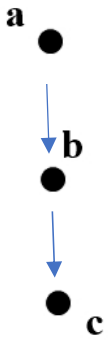
Genetic algorithms also allow hybridization, though we'll start simply with mutation. For hybridization, we start with a (random) group of network representations, find a couple of the best, and 'hybridize' them, picking a random spot and making a new network by sticking together the code left of that random point from the first network with the code right of that random point from the second one.

IV. In order to do what we've outlined here, we need a way to move from networks to 'adjacency codes': given a network, we can generate its code. Given a code, we can generate the network.

Another thing we need: a way to encode 'data' that is coming from the world. We can encode what happens in the world network when we 'initialize' various nodes, see what happens in our representation network with that same initialization, and then see how far off our representation is. That will allow us to choose between better and worse representations, which can in turn be used in picking some 'best' so far and mutating them to see if we get something that better matches the world.

V. So we ended up proposing a 'starting small' attempt at programming these basic components. These won't even be fully Bayesian nets, yet, with probabilities at nodes and conditional probabilities between them. Much simpler, for starters.

Let's let the world look like this:



We'll make it a purely deterministic causal network. If a happens, b is bound to happen. If b happens, c is bound to happen

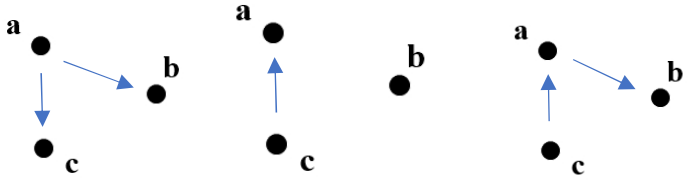
We'll also do it with time stamps: If a happens, *then* b happens, *then* c happens. If b happens, *then* c happens (and a doesn't).

That's the world. The data that comes from the world looks somehow like that—time-stamped data on what happens with initialization at each node. So maybe our data is:

If activate a then a minute later b then a minute later c
If activate b then a minute later c (but no a)
If activate c that's all you get.

Somehow we want to represent that 'data' and have our world network generate it. It's not yet clear how we should try to do that.

On the 'representation' side, we start with a few random networks. Maybe:

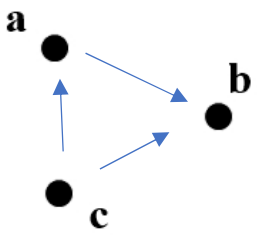


We ‘score’ them on how well they match the data. If we activate ‘a’ in our leftmost representation, b and c both happen on the next tick. Does that match the world? Well, sort of, maybe give it a score of .5. Activate ‘b’ in our representation and we don’t get ‘c’, as in the world. Score 0 on that test. Activate ‘c’ and we just get ‘c’. Right, so 1. Total score? Perhaps 1.5 out of 3 or something. Here again it’s not quite clear how we should do things.

We then pick our highest-scoring network, perhaps, encode it and generate additional mutations. We then test those, and so forth.

Ideally, we eventually stumble on a network that gets us a perfect score. We’ve figured out the world.

And of course we want to try the procedure on other worlds as well—for example



[We’re really interested in directed acyclic graphs. How we eliminate cycles, if we do, and how we interpret arrows both ways between two nodes are open questions.]

VI. The next step is to try programming this starting small thing, with just 3 nodes and just deterministic causality with a convenient time stamp.

Patrick (with wife Terri and Wolfgang the dog) is taking off on a road trip to Santa Fe over the next two weeks or so.

Amber and Dennis are going to take the lead, with Zhongming and Sophia as further resources, meeting once a week over the next two weeks.

I hope everyone will share ideas and points of progress with everyone in group emails.

Thanks,
Patrick