Abstract

A formalism called Sequential Input Graphical Model Assessment (SIGMA) diagrams is introduced. It shares with conventional graphical models features for specifying probabilistic inferences. In addition, it provides facilities for describing temporal patterns of evidence. The formalism was motivated by a need for transparent representations of assessment processes in the INFACT online learning environment. An editor and interpreter for SIGMA diagrams have been implemented and embedded in INFACT. The interpreter works with either stored event data, real-time event processing, or a combination of both. Examples diagrams are given, and the expressive power of the formalism is discussed.

1. Introduction

Online learning has the potential to add the following elements to students’ educational experience: connection to distant resources and people, presentation of active material such as simulations and animations, and high-quality assessment of progress using powerful computer methods. This paper focuses on a graphical means of expressing schemes for performing assessment in online learning.

Educational assessment is often performed using standardized, multiple-choice tests. This method is well understood and it can produce accurate measurements of student understanding or progress. However, such testing is often found to be obtrusive, and can induce detrimental levels of anxiety [5].

Our assessment methods are designed to be unobtrusive. They make use of data from records of the activity generated by students as they work online. Inferences are drawn from what students do.

In our previous research, we created an online learning environment called INFACT that makes it easy to experiment with collaborative learning in small groups and with new assessment methods. INFACT contains a forum component that supports small-group discussions and communication with sketches. INFACT stands for Integrated, Networked, Facet-based Assessment Capture Tool [7][8]. The assessment processes described by the SIGMA diagrams presented here run as agents in INFACT.

2. SIGMA Diagrams

In this section, we describe a progression of more and more powerful diagrams, ending in what we call Sequential Input Graphical Model Assessment (SIGMA) diagrams.

2.1. Bayes Nets

During the 1990s, great attention in the research community was paid to the design of Bayes networks for a wide variety of diagnosis tasks. These networks provide an efficient means for representing and computing the effects of various configurations of evidence on hypotheses. Given evidential propositions $E_1$, $E_2$, ..., $E_m$, and hypotheses $H_1$, $H_2$, ..., $H_n$, the probability of a particular hypothesis $H_k$ is a function of the probabilities of the evidence:

$$P(H_k) = f_k(P(E_1), P(E_2), \ldots, P(E_m))$$

The diagram in Figure 1 displays the inputs to $H_k$. In an educational assessment context, each node might represent a piece of evidence or a hypothesis about something the student might or might not believe about the subject. In our work, we assume that the $E_i$ are essentially true or false (or somewhere in between, when probabilities are taken into account); this is a
restriction, since Bayes nets often consider the $E_i$ to be
random variables with non-binary ranges. Also, we
assume that traditional Bayes nets are acyclic graphs
with probability updating proceeding directionally from
evidence nodes toward downstream hypothesis nodes.
For more on Bayes nets, see Pearl [3][4].

\[
P(N_k | E) = f(P(N_{k1}), P(N_{k2}), \ldots, P(N_{knk}))
\]

where $N_{k1}, \ldots, N_{knk}$ are the parents of $N_k$. Here nk
represents the number of parents of $N_k$.

Traditionally, this Bayes net structure tells only how
these nodes are interconnected and how to update
probabilities of downstream nodes when probability
values of top-level nodes (nodes without parents) are
altered by external agents. They can also be used in
reverse, using Bayes’ rule (e.g., to infer the probabilities
of causes from observed effects). They do not specify
how the top-level nodes respond to events or what
actions to take when bottom-level nodes reach high
posterior probability values. We have sought a single
structure that could encapsulate not only these
probabilistic relationships among propositional nodes
but also the additional information needed to have
operational specifications for educational assessment
processes. In principle, this could be accomplished by
adding complex labels to the nodes of the network. In a
way, that is what we have done; however, we have
extended the range of meanings of arcs themselves so
that temporal relationships can also be represented.
This is explained in sections below.

2.2. Conditions and Actions

In order to provide an integrated representation of
an educational assessment schema, we have extended
the traditional probabilistic inference net with facilities
for specification of (a) evidence, and (b) actions. Each
node of our networks is of one of three possible types:
input, middle, or output. As usual for Bayes nets, all
nodes are endowed with prior probabilities. Middle
nodes and output nodes contain representations of
functions that tell how to compute posterior
probabilities from probabilities of the parents.
However, going beyond traditional Bayes nets, we
endow the input nodes of our nets with expressions
that allow them to recognize the evidence to which they
must respond. Also, output nodes are endowed with
expressions that specify actions to take within the
INFAC automatic inference system whenever their posterior probabilities
exceed their thresholds.

An example of an input node condition is a tagged
regular expression that specifies a type of event to look
at (e.g., a mathematical-formula entry event) and a
pattern to look for in the events’ description field, for
example, any mention of the subexpression “2*x”. An
element of an output node action is to send an email
message to a student or store an achievement record in
INFAC.

Once an assessment designer (the “user”) has
created a network, it can be deployed within INFAC to
analyze student activity in real-time. As students
generate events, the input nodes of the network listen,
and they fire whenever their conditions are satisfied.
Firing generally means propagating probability updates
to children nodes and their children, etc. The networks
are automatically instantiated with one instance per
student, so that all the students in a class can be
independently assessed at the same time.

\[
A \rightarrow C \rightarrow D
\]

Fig. 2. A probabilistic network of the general form “If A
and B according to C then D.” Here A and B are input
nodes, C is a middle node, and D is an output node.

These nets, without any additional extensions are
useful in assessing some aspects of student learning.
Not only do we use them in our system INFAC, but
others have used straight Bayes nets for educational assessment [2]. However, the condition language for input nodes would have to be made substantially more complex if we wanted individual input nodes to be able to detect specific temporal sequences of events. In order to provide a means to specify temporal ordering constraints in patterns of evidence, we have created extensions to traditional inference nets with some attractive properties.

2.3. Accumulating Repeated Evidence

In typical usage of Bayes nets, each piece of evidence is considered to be either present or absent. There may be a neutral state, if the particular formalism allows it, in which a prior probability serves to represent uncertainty about whether or not the evidence is present. In an event-driven context, if the evidence is the occurrence of a particular kind of event, then the first time such an event occurs, the “current” probability for that evidence jumps from its prior value to 1.0 and remains at 1.0. It will never change after the first qualifying event occurs.

In our system, the occurrence of multiple events of the same type (for example, multiple formula calculation events) may indicate a greater degree of fluency with a particular feature than only a single event would indicate. Therefore, there can be a need to permit multiple occurrences of similar evidence to increase the probability that is propagated to child nodes. To meet this need, we have provided input nodes with the capability to increase their output probability when additional pieces of matching evidence are detected. In addition to having a condition expression, an input node has a “response method.” A response method can be empty, in which case it outputs a 1 for all time after the first matching piece of evidence arrives. It can also contain a clause of the form (REQUIRE n), meaning that it will not fire until the n\textsuperscript{th} piece of qualifying evidence has been detected. And finally, it may contain a clause of the form (ACCUM_ZENO 0.6) which indicates that each time the node fires, its output probability will be raised 0.6 of its difference from 1.0.

The two types of clauses can be combined. For example, an input node that responds to image-viewing ZOOM-IN events with the response method (REQUIRE 5)(ACCUM_ZENO 0.5) will not fire until the student has zoomed in on an image 5 times, and then it will raise the output probability from its prior value (set by the designer, say at 0.2) to half the distance from that value to 1 (in this case, by 0.5 \times 0.8 = 0.4 to 0.6). The next time that student zooms in, the probability will move up again (in our example, to 0.8), etc., never quite reaching the value 1.0.

This particular extension to our inference nets requires that each student's instance of any input node that uses either of these features have a count value associated with it. The mechanism permits accumulation of unlimited numbers of evidence occurrences with corresponding increases in posterior probabilities for downstream hypotheses. In allowing accumulation of evidence of the same kind over time, it adds a fundamentally different kind of capability to standard probabilistic inference networks such as Bayes nets. However, this mechanism still does not permit networks to be sensitive to the relative temporal order of occurrences of different kinds of events.

2.4. Relative Ordering Constraints

In a typical organized learning activity in our image processing course, each student works through an activity sheet one step at a time. The steps are ordered. However, students sometimes skip steps or go back to steps they have skipped. Sometimes it's important for them to perform the steps in order.

One way to automatically establish that a two-step sequence has been completed is to take evidence for completion of each of the two steps and combine it, effectively computing the probability of the conjunction of statements about the steps: “The student has completed step A and the student has
completed step B.” However, this probability value is the same regardless of whether the student completed A and then B or B then A.

In order to allow selective response based on the order in which evidence arrives, we have made a very simple but powerful extension to our probabilistic inference nets that we call “Enabled input nodes.” As mentioned in the previous section, each node of our nets is either an input node, middle node, or output node. Without the enabled-input-node extension, each input node always has zero parents. However, with the extension, any input node is allowed to have any number of parent nodes. If it has at least one parent, then it cannot test any event for its condition unless it is “enabled.” In order to be enabled, it must compute a probability value based on the values of its parents and its own update function (which it must now have, like middle nodes and output nodes), and that value must exceed the node’s threshold (which it must now have, just as output nodes must). If it is enabled, then it may respond to events just as an input node without parents would respond, testing the event to see if it satisfies its condition, and if so, invoking the updating process for its children.

The effect of this extension is to permit restrictions on the temporal ordering of events that will activate portions of the network. The enabling capability takes full advantage of the probabilistic updating procedure already implemented, so that the enabling is essentially like a new kind of action --- an internal action.

A probabilistic inference net that contains (a) the ability for input nodes to recognize simple events with regular expressions on textual descriptions, and (b) the enabling mechanism for input nodes, we call a “token-free sequential-input graphical model assessment diagram” or TF-SIGMA diagram for short.

In the next section, we describe an additional mechanism, one we call “tokens,” that increases the specificity possible for temporal patterns of evidence. We’ll show that neither of these extensions requires any increase in the amount of runtime memory needed, per student, to perform the assessment, over that required by a similar model without temporal constraints.

Fig. 4. An inference net with an ordering constraint. It has the form, “If A then B according to C then D.”

The use of enabled input nodes permits simple ordering constraints to be placed on sequences of events that will trigger responses by the Bayes network. However, it is difficult, if not impossible, to specify more complex sequences such as might be specified by a finite-state automaton or a Petri net. For example, we may wish to recognize a pattern of alternation between zoom-in and zoom-out events, and to require that 10 repetitions of the zoom-in/zoom-out sequence be detected before evidence of this pattern is propagated to other parts of the network.

We have implemented another extension to our augmented probabilistic inference nets with the twin goals of facilitating the expression of more complicated sequences of events and maintaining a low per-student memory cost in representing student progress.

Each input node of a student instance of the network is permitted to hold a “token.” Such a token resembles tokens in Petri nets. There can be at most one token at any input node. Tokens are all identical; there are no “colors” on tokens. The condition for an input node may contain a component that indicates that a token on a parent node is required for the current node to be enabled. This requirement is in addition to the computed probability exceeding a threshold, as per the enabled-input-node extension. When an enabled input node that requires a parent token fires, it grabs the token; the parent’s token is removed, and the current node gets a token. Such a transition is similar to a disjunctive transition in a Petri net.

It is easy to imagine how a token would be used to keep track of the alternating sequence of zoom-in and zoom-out events. There would be two input nodes, N₁ and N₂, with N₁ responding to zoom-in events and N₂ responding to zoom-out events. Each node would be a “parent” of the other. The initial placement of the token would be at N₂. When the first zoom-in event occurs, N₁ would be enabled by virtue of its parent having the token, and it would respond to the event by grabbing the token. N₂ would respond to the next zoom-out event. If N₂ has a response method of (REQUIRE 10),

Fig. 5. Variation in which evidence at A can have a direct effect at C in addition to enabling B to affect C.

2.5. Tokens
then it will not invoke an update on its downstream nodes until it has seen its 10th zoom-out event when enabled by a token.

We also allow additional restrictions in the condition to require that the token appear in a particular parent in conjunction with a particular subcondition on the event description. That makes it possible to have conditions of the form, “If parent 1 has the token and event A happens OR if parent 2 has the token and event B happens, then fire.” Finally, we permit a subcondition to be included that requires that all parents of the node have tokens. This makes it possible to require that multiple parent nodes be in the tokenized state as a prerequisite to further propagation of activity.

We have applied SIGMA diagrams within the INFACt environment to the automatic assessment of student learning in image processing activities. This is described in a companion paper [9].

In the future, we plan to embed SIGMA diagrams within a visualization facility to show the accumulation of evidence in real-time, as one effort to bring Bayesian analysis to the realm of “open learner models” [1].

4. Acknowledgements

Thanks to E. Hunt and D. Madigan for comments on the paper and literature referrals. The partial support of the US National Science Foundation under grants 0121345 and 0537322 is gratefully acknowledged.

5. References


![Fig. 6. A Sequential Input Graphical Model Assessment diagram in which three input nodes form a cycle. The initial marking has a token at C.](image)

The pattern of tokens within the input nodes of such a network is called its *marking*, following the terminology of Petri nets. When one of these extended nets is specified, the user gives an *initial marking*. There must be at least one token in the initial marking for the token mechanism to play any role in the updating of instances of the net. Similarly, there must be at least one input node whose condition requires the presence of a token at a parent.

3. Discussion

A SIGMA diagram is a computational structure that specifies an educational assessment scheme. A single SIGMA diagram does what otherwise might require a Bayes net, several finite-state automata, and a language for describing events. Part of the diagram is displayed as a network while its parameters are normally seen only in dialog boxes.

It is possible within the INFACt system to have multiple diagrams active at the same time for different (or for the same) activities. One diagram can be handling detailed analysis while another is making a coarse-grained summary of activity.