Incorrect Moves and Testable States

Dimiter Dobrev

Institute of Mathematics and Informatics, Bulgarian Academy of Sciences.
d@dobrev.com

Abstract

How do we describe the invisible? Let us take a sequence: input, output, input, output ... Behind this sequence stands a world and the sequence of its internal states. We do not see the internal state of the world, but only a part of it. To describe that part which is invisible, we will use the concept of 'incorrect move' and its generalization 'testable state'. Thus, we will reduce the problem of partial observability to the problem of full observability.

1 Introduction

Our first task in the field of Reinforcement learning is to describe the current state of the world (or the current position of the game). When we see everything (full observability) this question does not arise because in this case the input fully describes the state of the world. More interesting is the case when we see only a part of the world (partial observability). Then we will add more coordinates to the input vector, and thus we get a new vector which already fully describes the state of the world.

To add these new coordinates we will first introduce the concept of 'incorrect move'. The idea that not all moves are correct is not new. Other authors suggest, for example, that you cannot spend more money than you have. That is, they assume that the output is limited by a parameter that changes over time. What's new in this article is that we will use this parameter to give further description of the state of the world.

If at a specific moment we know what we see and what moves are correct at that specific moment, we know a lot about the current state of the world, yet we do not know everything. When we generalize the concept of 'incorrect move', we will derive the new concept of 'testable state'. If we add to the input vector the values of all testable states, we will get an infinite-dimensional vector that fully describes the state of the world. That is, for every two distinct states there will be a testable state whose value in these two states is different.

Incorrect moves and testable states are something that actually exists just like the input we get on each step. However, unlike the input, the value of testable states is not ready to derive. For example, is the door locked? To check this we need to push the handle and see whether the door will open, but we can do it only if we stand by the door. In other words, the check requires extra effort and it is not

always possible (there are moments in which we can check and moments in which we can't). The locked door can be considered as an example of both an incorrect move and of a testable state.

To describe the incorrect moves and testable states we will search for a theory that gives us this description. Of course, we may have many theories for a given testable state all of which will compete with each other over time.

What will the theory constitute? Statistics shows us that in specific situations a given testable state is true. Specific situations means a situation in which a conjunction is true. This conjunction may be associated only with the past, but may also be associated with the past and the future. For example, let's say we've checked and we've seen that a door is locked, but is it the door that we are interested in? We may decide that it is precisely this door on the basis of what we have seen before checking, or maybe after checking, a posteriori, we've seen something which tells us that it is exactly this door.

Another generalization of 'specific situations' will be to allow dependencies with memory (except dependencies without memory). A dependency without memory is the situation in which specific events occur in the course of several consecutive steps (i.e. this is an ordinary conjunction). A dependency with memory is the situation in which specific events occur at specific moments (not consecutive), and in the periods between those moments specific events do not occur.

The theory may be a set of such implications and this set can be generalized as a finite state machine. Let's take an example where we are moving in a castle in which some of the doors are locked. We can have a rule of the following type: "If you see a white sofa and you turn right, then the door will be locked." If we represent the map of the castle as a finite state machine, we will see that if after the white sofa we turn right, we are at the door X and that this door is locked. If we know the finite state machine, we can easily derive the corresponding rules. Unfortunately, we need the opposite – to derive a finite state machine from the rules, and that's a much more difficult task.

Let us now imagine a castle the doors of which are not permanently locked or unlocked but change following specific rules. Then our theory will suggest some sustainability of testable states. For example, if a door has been locked at a specific moment and shortly thereafter we check again, we assume that it will be locked again, especially if during this time no specific events have occurred (for example, to hear a click of door locks).

The next upgrade of the theory would be to assume that there is some kind of creature inside the castle, which unlocks and locks the doors in its sole discretion. Then we can predict whether a door is locked or unlocked predicting the behavior of that creature.

Once we've created one or several theories that predict a testable state, we will use these theories and gather statistics that will help us predict the future and to create new theories of other testable states. For example, in our case, if we've noticed that behind the locked door there is a princess, and a tiger behind the unlocked door, then based on the theory that tells us which door is locked, we can make a theory that tells us where the princesses are.

2 Definitions

Let's take a sequence of *output, input, output, input, …*, and the goal is to understand this sequence.

Of course, this sequence is not accidental. We can assume that we are playing a game and that's the sequence:

move, observation, move, observation ...

And our goal is to understand the rules of the game and what is the current position on the game board.

We might assume that we are in a world and then the sequence would be:

action, view, action, view ...

In this case, our goal is to understand the rules of this world and what is the current state of the world.

The description of the world is given by the functions *World* and *View*, and the following applies:

$s_{i+1}=World(s_i , a_{i+1})$

$v_i=View(s_i)$

Here, actions and observations (a_i and v_i) are vectors from scalars with dimensions n and m, respectively.

Our goal is to present the internal state (s_i) also as a vector of scalars, in which the first m coordinates will be exactly v_i. In other words, the function *View* will be the projective function that returns the first m coordinates of s_i.

We will call 'states' to all coordinates of s_i. We will call the first m ones 'visible states'. Other coordinates will be called 'testable states'.

The coordinates of the input and output will be called 'signals'. These are functions of time that return a scalar. To the input and output signals we will add other signals as well, like the testable states; more precisely – the value of the testable state according to the relevant theory, because we will not know the exact value of these states, and will approximate them with some theories. For each finite state machine we will add a signal whose value will be equal to the state in which the finite state machine is situated at the moment t. Of course, if the machine is nondeterministic, the value of this signal may not be exact but approximate.

We will call the Boolean signals 'events'. When the Boolean signal is truth, we will say that the event is happening.

3 Incorrect move

Shall we allow the existence of incorrect moves?

Our first suggestion is to choose a world in which all moves are correct. In the chess game, we cannot move the bishop as a knight, so it is natural to assume that some of our moves are incorrect or impossible.

Our second suggestion is to have incorrect moves and when we play such a move to assume that the world penalizes us with a loss. So we will very quickly learn to avoid incorrect moves, but here we are denied the opportunity to study the world by checking which move is correct (like touching the walls in the darkness).

Our third suggestion is: When an incorrect move is made the state of the world to remain the same, and instead of 'loss' the immediate reward to be 'incorrect move' and all other coordinates at the input to return 'nothing'. This option is better, but thus we would unnecessarily prolong the history. (We will call 'history' the following sequence: $a_1, v_1, ... , a_{t-1}, v_{t-1}$, where t is the current time). Given that the state of the world remains the same, there is no need to increase the count of the steps.

The fourth suggestion is: When you play an incorrect move, you to be informed that the move is incorrect but without prolongation of the history. The new history will look like this: $u_1, a_1, v_1, ... , u_t, a_t, v_t$. Here u_i is the set of incorrect moves at the i-th step which we have tried and we know for sure that they are incorrect. Thus the history is not prolonged, yet the information that certain moves are incorrect is recorded in the history. Here we assume that the order in which we've tried the incorrect moves does not matter.

The fifth option, which we will discuss in this article, will be even more liberal. In the previous suggestion we have the opportunity to try whether a move is incorrect, but if it proves correct, we have already made this move. Now we will assume that we can try different moves as many as we want and we can get information whether it is correct or incorrect for each one of them. After that, we

make a move, for which we already know that it is correct. We know because we've tried it. Here, the history is the same as with suggestion No. 5, except that u_i is a tuple of two sets, the first of which is from the tried incorrect moves, the second – from the tried correct moves.

There is a sixth option and it is for every step to get full information about which moves are correct and which are not. In other words, we get u_i with all the moves in it like they have been tested. However, we do not like this option because u_i may be too large, i.e. it may contain too much information. Moreover, we assume that after some training we will have a fairly accurate idea about which move is correct and which is incorrect and will not have to make many tests in order to understand this.

4 Testable states

A testable state is the result of an experiment. Here are two examples:

The door is locked
If I push the handle ⇒ the door will open.

The stove is hot
If I touch the stove ⇒ I will get burned.

Here we have the testable state and the experiment, which must be made in order to obtain the value of the testable state. The result of the experiment can be 'Yes' or 'No'.

From the above two examples you might get the impression that the experiment consists of any actions that we need to do, but it is not so. The experiment could be something that happens without our intervention. Here is an example:

The roof is damaged
If it rains ⇒ the roof will leak.

In this example the testable state is checked when it rains, and this is something that does not depend on our actions.

Definition: A testable state will consist of two statements. We will call the first one 'condition' and the second one 'conclusion'. Below we will give additional requirements to the condition and the conclusion of a testable state.

Definition: We say that a testable state is a special case of another one if they have the same conclusion, and if the condition of the second is a special case of the condition of the first.

Let's define incorrect move.

Definition: An incorrect move is testable state whose condition is a conjunction in which all literals are from the output and are for the moment $t+1$ and whose conclusion would be 'the move is incorrect'.

That is, the incorrect move is testable state, whose condition is a cumulative move (at the moment $t+1$).

To complete the definition of testable state we will first define an immediately testable state is. This is testable state that, if it can be tested, the test will be carried out at the next step.

Definition: An immediately testable state is one of the following:

1. An incorrect move.

2. A testable state, whose condition is a conjunction, in which all literals are for the moment $t+1$, and the conclusion would be a literal from the input, also for the moment $t+1$.

Here two questions arise. Why the conclusion is only one literal and why the literal is from the input? The answer to the first question will be: If the conclusion is the conjunction of two literals, then we can present this testable state by four other testable states in which the conclusion is only one literal. If this testable state is of the type $A \Rightarrow B_1 \,\&\, B_2$, this testable state is a truth just when the testable states $A \Rightarrow B_1$ and $A \Rightarrow B_2$ are a truth. This state is a lie just when the testable states $A \,\&\, B_1 \Rightarrow \neg B_2$ and $A \,\&\, B_2 \Rightarrow \neg B_1$ are a truth. (Here the negation with Boolean signals is one literal, and with non-Boolean ones – a disjunction of several literals.)

As for the second question: We want to describe the world, not to describe our behavior. That in certain circumstances we have always played something (or have never played this thing) is not a description of the world, and a description of our behavior. We may never have played it because it's an incorrect move or we could have always played it, because this is the only correct move. This latter is already a description of the state of the world, but this information is given by the term 'incorrect move'.

Having defined an immediately testable state we are to get a definition of the term 'testable state" by adding something to the condition. We will add precondition or postcondition or both. A precondition will be a conjunction of literals which are for moments before $t+1$, and a postcondition will be a conjunction of literals which are for moments after $t+1$.

Here, one part of the conjunction is associated with the past (before $t+1$), and the other part is associated with the future (after t). The testable state is a prediction for the result of carrying out an experiment that has yet to be held. So a testable state must fully involve the future. Otherwise, we get something that is partially known and it can not be verified (for the known part is a lie) or will be a testable state, but some other testable state (the first will be a special case of the second) because the known part will be a truth and the new testable state will depend only on the part that is still unknown.

Therefore we will correct the definition of a testable state by shifting the entire conjunction to the right, so it all goes into the future (we add a k to the time of all literals). We need to shift the conjunction to the right until the most left literal is for the moment $t+1$.

If we shift further to the right (e.g. by one more step), we will get another testable state, but it will be a prediction of more distant future. So this new testable state will be a truth if the old testable state is a truth at step $t+1$ whatever happened at step $t+1$. That is, the new testable state will be a more common case of the old taken for step $t+1$. This is so because the new one can be obtained from the old one taken for step $t+1$ by adding what happened at step t (i.e. to add a conjunction describing completely the output at step t). That is, we quite naturally get that the testable state that looks ahead to the future, is a more general case of the one that looks closer.

In the definition of a testable state we can assume that the precondition and the postcondition are not conjunctions but cumulative conjunctions. That is, the precondition and the postcondition can have memory and apply for a longer period of time (and not just for a few steps).

Note that the number of immediately testable states is final (if the input and output signals are finite functions). By adding preconditions and postconditions we get an infinite number of testable states, which is natural, because we can not describe an infinite world with finitely many parameters.

5 Conclusion

In the field of AI our first task is to say what we are looking for, and the second task is to find the thing we are looking for. This article is entirely devoted to the second task.

Articles [Dobrev, 2000, 2005, 2013], which are devoted to the first task, tell us that the thing we are looking for is an intelligent program that proves its intelligence by demonstrating it can understand an unknown world and cope in it to a sufficient degree.

The key element in this article is the understanding of the world itself, and more precisely – the understanding of the state of the world. We argue that if we understand the state of the world, we understand the world itself. Formally speaking, to understand the state of the world is like reducing the problem for Reinforcement learning with partial observability to the problem for Reinforcement learning with full observability. Not accidentally, almost all articles on Reinforcement learning deal with the case of full observability. This is because it is the easiest case. The only problem in this case is to overcome the combinatorial explosion. Of course, combinatorial explosion itself is a big enough problem because we get algorithms that would work if you have an endlessly fast computer and indefinitely long time for training (long in terms of the number of steps). Such algorithms are completely useless in practice and their value is only theoretical.

In this article we presented the state of the world as infinite-dimensional vector of the values of all testable states. This seems enough to understand the state of the world, but we need a finite description and we would like this description to be as simple as possible. For this purpose, we've made three additional steps. A model of the world was introduced as a finite state machine. Each such machine describes an infinite number of testable states and this is a very simple description which is easy to operate.

The second step was the assumption that testable states are inert and change only if specific events occur. That is, if between two checks, none of these events has occurred, we can assume that the value of the testable state has not changed.

The third step was the introduction of agents which under certain conditions may alter the values of testable states. This step is particularly important because the agent hides in itself much of the complexity of the world and thus the world becomes much simpler and more understandable. We will not describe the agent as a system of formal rules. Instead, we will approximate it with assumptions such as that it is our ally and tries to help us or that it is an opponent and tries to counteract.

Without these three steps the understanding of a more complex world would be completely impossible. Formally speaking, testable states only would be sufficient to understand the state of the world. Even testable states which conditions are simple conjunction dependent only on the last few steps of the past are sufficient. However, without the additional generalizations made, we would face an enormous combinatorial explosion.

References

Dobrev D. (2000, November) AI – What is this. In *PC Magazine – Bulgaria, pp.12-13.* http://www.dobrev.com/AI/definition.html

Dobrev D. (2005) Formal Definition of Artificial Intelligence, *International Journal "Information Theories & Applications", vol.12, Number 3, 2005, pp.277-285.* http://www.dobrev.com/AI/

Dobrev D. (2013, January) Comparison between the two definitions of AI. *arXiv:1302.0216.* http://www.dobrev.com/AI/