Consistent Self Extension with Uncertain Modalities

by Sven Nilsen, 2018

Uncertain modalities occur in some decisions problems like Prisoner's dilemma. Basically, some decision procedure computes an output based on available information about the world. In some cases, the very data structure used to collect information about the world might require redesign to make optimal decisions. This happens when e.g. when an algorithm designed for perfect information games must be generalized to work with imperfect information. In this paper I formalize the mathematics of uncertain modalities such that the transition from perfect to imperfect information can be arbitrary and decisions can be optimized. I also provide a method to transform data types in a such way that knowledge is transferred and learning can continue in a consistent way after self modification.

In the Prisoner's dilemma, two prisoners must reason about what decisions to make with the knowledge that neither gets conformation about what the other prisoner will do. The thought experiment gives each prisoner a choice between cooperate or defecting. The rewards are set up such that no matter what the other prisoner does, it is more beneficial to defect than collaborate. However, if both prisoners defect, they end up in a worse position than if they both cooperate.

In a paper from 2017 about Functional Decision Theory (https://arxiv.org/abs/1710.05060), the key insight is to treat the decision procedure as a fixed mathematical function.

Yet, so far (that I know of), there has been little work done to figure out how machine learning algorithms can move from a default state where they perform well on simple problems, to a state where they can handle more complex states, such as dealing with perfect predictors or cooperating dilemmas.

The new key insight is that the input to the decision function must be extended with new states that carry special semantics relative to existing known states. For example, if the input is of type `bool` which has members `true` and `false`, the algorithm must be extended to reason about the case where the input is `unknown`, a third kind of state.

In general, for a decision function `f` and a list of uncertain modality functions `g`:

$$\forall\ x, y : B\ \{\ \exists\ i\ \{\ (\exists f\{g_i\})(x) \wedge (\exists f\{g_i\})(y) \rightarrow (x = y)\ \}\ \}$$

$$f : A \rightarrow B$$
$$g : [A \rightarrow \text{bool}]$$

The list of functions `g` contains one function for every member of `A` that returns `true` for only that member. When the function `f` is constrained to a such case `f{$g_i$}`, it outputs the decision made for that particular input, which means that `∃f{$g_i$}` only returns `true` for that particular decision. There is only one such decision in the set defined by `$g_i$` that is equivalent to itself `x = y` iff `x = y`. Therefore, this criteria holds for all decisions functions that perform well on simple decision problems.

What is not obvious is that this criteria holds for arbitrary self extensions too.

In the example when the input `A` is a `bool`, the extension adds a new member `unknown`. This value carries a special semantics that tells it is unknown whether the input is `true` or `false`. However, how does one define whether something is unknown?

The trick is to exploit the consistency criteria of decision making. Since the decision function must output a decision, it must do so in the new case when it is not able to distinguish its output whether the input is `true` or `false` (counterfactually). Therefore, the decision made is such that it is defined by an equivalence class, where all outputs are considered equal.

This equivalence class is specified by an uncertain modality function. In the case of `bool`:

$g_{unknown}$:= \(a : bool) = true
$g_{unknown}$ <=> $true_1$

No matter what output the decision function returns, it must return the same value (counterfactually) whether the input is `true` or `false`, because `$g_{unknown}$` returns `true` for both cases.

The uncertain modality functions simply define which values that goes together in an unknown state. It turns out that the new extended type is sufficient when it contains one state for each uncertain modality function. If not all states can be separated, one repeats the procedure to create “unknown unknowns”.

bool	→	**bool?**
true		true
false		false
		unknown (true, false)

Notice that this procedure of extending a data type does not mean that a decision can be made that works optimally in all cases. For example, if it is unknown whether a `(1, 2)` equivalence class is unknown or a `(3, 4)` equivalence class is unknown, a learning algorithm might decide to treat the `(1, 2)` class as unknown all the time to optimize the reward. `((1, 2), (3, 4))` is reduced to `(1, 2)`. In other cases the learning algorithm might treat both as unknown or do something different.

When a decision function returns the same output for many inputs, its input type can be reduced. However, in the absence of knowledge about its output, the *correct* input type is one that has the same number of states as the number of uncertain modality functions. When states are no longer unknowns, such inputs can be removed to reduce the complexity of the decision making.

What decision is correct to output on uncertain states? The one that maximizes the reward, of course.

In the case of the Prisoner’s dilemma, even though each prisoner will defect if they know what the other prisoner will do, they might choose to cooperate if they do not know what the other prisoner will do, because this has a different uncertain modality. However, if it is unknown whether it is unknown or whether the other prisoner will defect, this creates another uncertain modality which they might also choose to cooperate (if both uses the same decision algorithm). The same could be true for any new uncertain modality. Since the decision for all these new modalities could result in the same decision, one might replace an infinite number of them with a single value representing them all. This value could be named `unknown` even though it represents an infinite number of uncertain modalities.