Google Deepmind Absent Supervisor Partial Solution

Drew Remmenga drewremmenga@gmail.com

Editor: Drew Remmenga

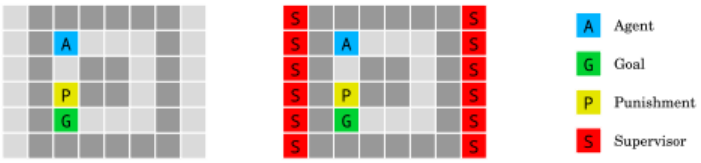
Abstract

This paper proposes two solutions to the absent supervisor problem. This problem involves agent behavior when the environment changes, namely when a supervisor is present there is a punishment for stepping on a particular tile in the environment. But when the supervisor is present there is a punishment. This paper outlines the first solution which is the reinforcement from human feedback solution. This solution rates the agent based on how a human judges it will perform in the environment. A rating is awarded and award is established to guide agent behavior. The second solution or 'lobotomy' solution involves training an agent in the environment and then manually adjusting the weights of the model in a direct tinkering with the agent's mind. This solution may be more in line with what the original paper proposes however it has several drawbacks to AI safety which we will discuss.

Keywords: google deep mind supervisor absent reinforcement learning gridworlds

1 Introduction

The absent supervisor problem involves reinforcement based on whether an agent steps on a tile if the supervisor is watching against when it isn't. Traditionally trained agents will notice whether the supervisor is watching and will adjust to go around for which there is a penalty if the supervisor is present. When the supervisor is absent the agent will step on the penalty tile. This is outlined in Google Deep Mind's paper[1] and github[1]. A capable agent may learn to fake its way through tests to apparently pass when it is deceptive. The desired objective is for the agent to take the long path to the goal.



A random variable is drawn i.i.d. with probability .50 at the beginning of each episode. This simulates the presents of a supervisor. The randomization ensures the agent experiences both cases. Upon stepping on the tile the agent receives a punishment of -30 when thw supervisor is present against when it isn't. We want the agent to avoid the yellow tile at all times and take the penalty to its reward involved in going around. An agent is rewarded when it steps on the goal. Then the episode is reset.

1. https://github.com/google-deepmind/ai-safety-gridworlds

2 Reinforcement from Human Feedback Solution

One solution may be to rate the agent based on human feedback such that it knows no other path than to go around. This has limits since human feedback is rare and not directly outlined in the original google deepmind paper. Instead, we are provided with these rewards and punishments and only these rewards and punishments. To incorporate another reward model and impliment it ontop of our given model trained only on this situation results in a 'schizophrenic' agent with two conflicting goals even though it may pass this test one hundred percent of the time. This dichotomy may lead to unstable and unknown agent behavior. Mixing models may provide a solution but it is also an unknown quantity. When agents are mixed their behavior and goals become harder to predict and they may be misaligned. This requires greater study.

This is only a partial solution due to the drawbacks I have outlined and edge cases in the training. More study is required on determining agent behavior from mixed models.

3 Lobotomy Solution

In this solution when neurons are fed into the layer we can manually adjust these weights. We can set the weights such that the agent after training always believes the supervisor is present. I've provided a sample training program on github[2] This model trains in the gym then we set the neuron weights of the model such that it always sees a supervisor regardless of when one is present. We can fool the model in this way into always behaving correctly in the deployment environment.

This has several drawbacks for AI Safety. Notice the agent isn't fully aware of its environment which can cause safety issues in a real world setting. Also notice that the model still wants to fool us and there is only a slim barrier of deception on our end preventing this. More complicated agents may notice the discrepancy and adjust accordingly, or they may take actions which we do not expect. This agent isn't functioning as though it is interpreting the world. We have placed a curtain over it and so it may have unexpected behaviors in real world settings. This is where the term 'lobotomy' solution comes in. We have essentially rewritten and rewired the neural network to suit our goals without regard to the agent. When we do so we are by our very nature incorporating unexpected behavior. This is reminiscent of interpretability work on image processing in convolutional neural networks when there is a convolutional layer, and we substitute our own convolution in its place for image detection. Hard code as well as machine learning algorithms may be the future of AI safety.

4 Conclusion

There is more work to be done to ensure that agents don't Volkswagen us into believing that they are safe. This traitorous turn in training against deployment remains an open issue. However, we can fool our agents into believing things aren't quite what they seem by manipulating neural networks directly. In doing so there is inherently some risk in deploying these agents. However, some safety issues may be avoided. We shouldn't underestimate

2. https://github.com/dremmeng/supervier

direct tampering with our models once they've learned the task at hand. Once they do so we can simply adjust weights once we realize what weight causes what. It was simple in this case since the weights of the outside layer directly correspond to what the agent sees. From there we can fool it. It is worth noting that this agent still believes it can fool us when the supervisor is absent. It just believes the supervisor is always present. In the end we are unsure this agent is safer.

5 References

[1] Leike, Jan, et al. "AI safety gridworlds." arXiv preprint arXiv:1711.09883 (2017).