Visão Geral do Projeto

Neste projeto você aplicará técnicas de aprendizagem por reforço para um agente auto-condutor em um mundo simplificado, auxiliando-o a chegar de maneira efetiva até seus destinos em um tempo atribuído. Primeiro você investigará o ambiente em que o agente opera implementando um algoritmo de direção bastante básico. Uma vez que o agente foi bem sucedido ao operar dentro do ambiente, você então identificará cada possível estado que o agente pode estar, considerando situações como semáforo e tráfego de sentido contrário em cada intersecção. Com os estados identificados, você implementará um algoritmo Q-Learning para o agente auto-condutor ser guiado até seu destino dentro do tempo estimado. Por fim, você irá melhorar o algoritmo Q-Learning no qual você já trabalhou para encontrar a melhor configuração de aprendizagem e fatores de exploração que asseguram que o agente auto-condutor chegue ao seu destino com resultados positivos consistentemente.

Descrição

Em um futuro não tão distante, empresas de táxi dos Estados Unidos não mais contratarão condutores humanos para operar sua frota de veículos. Em vez disso, os táxis serão operados por agentes auto-condutores - conhecido como táxi-inteligente -, transportando pessoas de um local para outro dentro das cidades que essas empresas operam. Na maiorias das áreas metropolitanas, como Chicago, Cidade de Nova Iorque e São Francisco, um crescente número de pessoas começou a confiar nos táxis-inteligentes para chegar onde precisam ir da maneira mais segura e eficiente possível. Embora táxis-inteligentes tenham se tornado o transporte mais utilizado, surgem preocupações de que um agente auto-condutor talvez não seja seguro ou eficiente como os condutores humanos, particularmente quando considerado os semáforos da cidade ou outros veículos. Para aliviar essa situação, sua tarefa como funcionário de uma companhia nacional de táxis é usar técnicas de aprendizagem por reforço para desenvolver uma demonstração de operação dos táxis-inteligentes em tempo real para provar que ambas segurança e eficiência podem ser alcançados.

Requisitos de Software

Este projeto usará o seguinte software e bibliotecas de Python:

- Python 2.7
- NumPy
- PyGame
    - Links auxiliares para instalar o PyGame:
    - Começando

- Informações sobre o PyGame
- Grupo do Google
- Subreddit do PyGame

Caso você não tenha o Python instalado ainda, é altamente recomendado que você instale a distribuição Anaconda de Python, que já tem os pacotes acima e mais inclusos. Tenha certeza que você selecionou o instalador do Python 2.7 e não o instalador do Python 3.x. O `pygame` pode então ser instalando usando um dos seguintes comandos:

Mac:

```
conda install -c https://conda.anaconda.org/quasiben pygame
```

Windows e Linux:

```
conda install -c https://conda.anaconda.org/tlatorre pygame
```

Começando o Projeto

Para essa tarefa, você encontrará o arquivo `smartcab.zip` que contém os arquivos necessários do projeto para download na seção Recursos. *Você também pode visitar nosso GitHub de projetos de Machine Learning para ter acesso a todos os projetos disponíveis para este Nanodegree.*

Este projeto contém dois diretórios:

- `/images/`: Essa pasta contém várias imagens de carros que podem ser utilizadas na interface gráfica do usuário. Você não precisará modificar ou criar qualquer arquivo nesse diretório.
- `/smartcab/`: Essa pasta contém os scripts de Python que criarão o ambiente, a interface gráfica do usuário, a simulação e os agentes. Você não precisará modificar ou criar qualquer arquivo nesse diretório, exceto pelo `agent.py`.

No `/smartcab/` há os seguintes quatro arquivos:

- Modifique:
    - `agent.py` Esse é o arquivo principal de Python em que você irá executar seu trabalho deste projeto.

- Não modifique:
    - `environment.py`: Esse arquivo de Python criará o ambiente do táxi-inteligente.

- `planner.py`: Esse arquivo Python criará um planejador de alto nível para que o agente siga em direção de um objetivo definido.

- `simulation.py`: Esse arquivo Python criará a simulação e a interface gráfica do usuário.

Compilando o Código

No terminal ou na janela de comando, navegue até o nível superior do diretório do projeto`smartcab/` (que contém os dois diretórios do projeto) e execute os seguintes comandos:

`python smartcab/agent.py` ou

`python -m smartcab.agent`

Isso irá executar o arquivo `agent.py` e o seu código do agente implementado dentro do ambiente. O arquivo README também foi incluído com os arquivos do projeto, que contém informações necessárias adicionais ou instruções para o projeto. Os seguintes slides Definições e Tarefas trarão detalhes de como o projeto será completado.

Definições

Ambiente

O táxi-inteligente opera em uma cidade ideal, em forma de malha (similar a cidade de Nova Iorque), com estradas indo nas direções Norte-Sul e Leste-Oeste. Outros veículos certamente estarão presentes na estrada, mas não haverá pedestres para se preocupar. A cada intersecção há um semáforo que permite tráfego tanto na direção Norte-Sul ou direção Leste-Oeste. Regras norte-americanas de trânsito de estradas aplicadas:

- No sinal verde, curva à esquerda é permitido se não houver nenhum tráfego no sentido contrário fazendo curva à direita ou indo reto pela intersecção.
- No sinal vermelho, curva à direita é permitido se não houver nenhum tráfego no sentido contrário se aproximando da sua esquerda pela intersecção. Para entender como conduzir corretamente o tráfego de sentido contrário ao virar à esquerda, você pode assistir a esse vídeo oficial sobre educação de condutores ou essa palestra calorosa.

Entradas e Saídas

Assuma que é atribuído ao táxi-inteligente um plano de rota baseado na localização inicial e o destino do passageiro. A rota é dividida a cada intersecção em um ponto de navegação, e você pode assumir que o táxi-inteligente, a qualquer instante, está em alguma intersecção no mundo. Portanto, o próximo ponto de navegação para o destino, assumindo que o destino ainda não foi alcançado, é uma intersecção distante em uma direção (Norte, Sul, Leste ou Oeste). O táxi-inteligente tem apenas uma visão egocêntrica da intersecção que é a: Ele pode determinar o estado do semáforo adequado para a sua direção e se há um veículo na intersecção para cada uma das direções contrárias. Para cada ação, o táxi-inteligente pode ficar ocioso na intersecção ou dirigir para a próxima

intersecção à esquerda, à direita ou em frente. Por último, cada viagem tem um tempo para alcançar o destino que diminui a cada ação tomada (os passageiros querem chegar no seu destino rapidamente). Se o tempo atribuído torna-se zero antes de alcançar o destino, a viagem falhou.

Recompensas e Objetivos

O táxi-inteligente recebe uma recompensa a cada viagem completa bem-sucedida, e também recebe recompensas menores por cada ação que ele executa com sucesso que obedeça as leis de trânsito. O táxi-inteligente recebe pequenas penalidade para cada ação incorreta, e grandes penalidades para cada ação que viole as leis de trânsito ou que cause um acidente com outro veículo. Baseado nas recompensas e penalidade que o táxi-inteligente recebe, a implementação do agente auto-condutor deverá aprender um política ótima para dirigir nas estradas da cidade enquanto obedece as leis de trânsito, evitando acidentes e alcançando os destinos dos passageiros no tempo atribuído.

Tarefas

Relatório do Projeto

Será necessário que você submeta um relatório do projeto junto de seu código de agente modificado como parte da sua submissão. Assim que você completar as tarefas abaixo, inclua respostas completas e detalhadas para cada questão *fornecida em itálico*.

Implementar um Agente Condutor Básico

Para começar, sua única tarefa é fazer o táxi-inteligente movimentar-se ao redor do ambiente. Nesse momento, você não deverá se preocupar com nenhum política de optimização de condução. Note que o agente condutor está dando as seguintes informações a cada intersecção:

- O próximo ponto de navegação é relativo a sua própria localização e direção.
- O estado do semáforo na intersecção e a presença de veículos em direção contrária vindo de outras direções.
- O atual tempo que sobrou do prazo estipulado.

Para completar essa tarefa, simplesmente faça com que seu agente condutor escolha uma ação qualquer do conjunto de ações possíveis (`None`, `'forward'`, `'left'`, `'direita'`) a cada intersecção, desconsiderando a informação de entrada acima. Defina a simulação de execução do prazo final, `enforce_deadline` para `False` e observa como ele executa.

PERGUNTAS: Observe o que você vê do comportamento do agente ao realizar ações aleatórias. O táxi-inteligente eventualmente chega ao seu destino? Há outras observações interessantes a serem feitas?

Informar o Agente Condutor

Agora que seu agente condutor é capaz de ser movimentar pelo ambiente, sua próxima tarefa é identificar um conjunto de estados que são apropriados para a modelagem do táxi-inteligente e o ambiente. A fonte principal de variáveis de estado são as atuais entradas na intersecção, mas nem todas podem precisar de representação. Você pode optar por definir explicitamente os estados ou usar algumas combinações de entrada como um estado implícito. A cada passo de tempo, processe as entradas e atualize o estado atual do agente utilizando a variável `self.state`. Continue com a obrigação do prazo da simulação `enforce_deadline` sendo ajustado para `False` e observe como seu agente condutor agora reporta a mudança de estado ao progredir da simulação.

PERGUNTAS: Quais estados definidos por você são apropriados para modelar o táxi-inteligente e o ambiente? Por que você acredita que cada um desses estados são apropriados para esse problema?

OPCIONAL: Quantos estados no total existem para o táxi-inteligente nesse ambiente? Esse número parece razoável dado que o objetivo do Q-Learning é aprender e tomar decisões informadas sobre cada estado? Por que e por que não?

Implementar um Agente Condutor Q-Learning

Com o seu agente condutor bem capacitado para interpretar as informações de entrada e ter um mapeamento dos estados do ambiente, sua próxima tarefa é implementar um algoritmo Q-Learning para o seu agente condutor escolher a *melhor* ação a cada passo, baseado nos Q-values do estado e ação atuais. Cada

ação tomada pelo táxi-inteligente vai produzir uma recompensa que dependerá do estado do ambiente. O agente condutor Q-Learning vai precisar considerar essas recompensas ao atualizar os Q-values. Uma vez implementado, ajuste a obrigação do prazo da simulação `enforce_deadline` para `True`. Execute a simulação e observe como os táxis-inteligentes se movimentam no ambiente em cada tentativa.

As fórmulas para atualizar os Q-values podem ser encontrados neste vídeo.

PERGUNTAS: Quais mudanças você percebe no comportamento do agente quando comparado ao agente condutor básico quando ações aleatórias são sempre tomadas? Por que esse comportamente está acontecendo?

Melhorar o Agente Condutor Q-Learning

Sua última tarefa para este projeto é melhorar seu agente condutor até que, depois de treinado o suficiente, o táxi-inteligente seja capaz de alcançar o destino dentro do tempo alocado de maneira segura e eficiente. Parâmetros do algoritmo do Q-Learning, como a taxa de aprendizagem (`alpha`), o fator de desconto (`gamma`) e a taxa de exploração (`epsilon`), todos contribuem para a habilidade do agente condutor de aprender a melhor ação para cada estado. Para melhorar o sucesso do seu táxi-inteligente:

- Ajuste o número de tentativas, `n_trials`, na simulação para 100.
- Execute a simulação com obrigação do prazo `enforce_deadline`, ajuste para `True` (você vai precisar reduzir o atraso da atualização `update_delay` e ajuste o `display` para `Falso`).
- Observe a aprendizagem do agente condutor e a taxa de sucesso do táxi-inteligente particularmente durante os últimos testes.

- Ajuste um ou vários dos parâmetros acima e itere esse processo.

Esta tarefa estará completa uma vez que você tenha chegado naquilo que você determinou como melhor combinação de parâmetros requisitados para que o agente condutor aprenda com sucesso.

PERGUNTAS: Reporte os valores diferente para os parâmetros sintonizados em sua implementação básica de Q-Learning. Para quais conjuntos de parâmetros o agente melhor se desempenha? Quão bom é o último desempenho do agente condutor?

PERGUNTAS: Seu agente se aproxima de encontrar um política ótima, por exemplo, chegar ao destino no tempo mínimo possível e sem incorrer nenhuma penalidade? Como você descreveria uma política ótima para este problema?

Submetendo o Projeto

Avaliação

Seu projeto será avaliado por um revisor da Udacity de acordo com a rubrica de projeto Treine um Táxi-Inteligente para Dirigir. Tenha certeza de ler a rubrica cuidadosamente e autoavaliar-se antes de submeter o projeto. Todos os critérios nessa rubrica devem *atender às especificações* para que você passe.

Arquivos para Submissão

Quando você estiver pronto para submeter seu projeto, reúna os seguintes arquivos e os comprima em um único arquivo para upload. Outra maneira seria fornecer os seguintes arquivos no seu repositório no GitHub em uma pasta nomeada `smartcab` para facilitar o acesso:

- O arquivo Python `agent.py` com todos os códigos implementados como requisitado nas instruções das tarefas.
- Um relatório do projeto em PDF com o nome report.pdf que responda todas as questões relacionadas as tarefas completadas. Esse arquivo *tem* que estar presente para que seu projeto seja avaliado.

Uma vez reunidos esses arquivos e que você tenha lido a rubrica do projeto, vá até a página de submissão de projeto.

PARTE 10

PRAZO DE ENTREGA

20/3/2017

Envio de Projeto

Em um futuro não tão distante, empresas de táxi dos Estados Unidos não mais contratarão condutores humanos para operar sua frota de veículos. Em vez disso, os táxis serão operados por agentes auto-condutores - conhecido como táxi-inteligente -, transportando pessoas de um local para outro dentro das cidades que essas empresas operam. Na maiorias das áreas metropolitanas, como Chicago, Cidade de Nova Iorque e São Francisco, um crescente número de pessoas começou a confiar nos táxis-inteligentes para chegar onde precisam ir da maneira mais segura e eficiente possível. Embora táxis-inteligentes tenham se tornado o transporte mais utilizado, surgem preocupações de que um agente auto-condutor talvez não seja seguro ou eficiente como os condutores humanos, particularmente quando considerado os semáforos da cidade ou outros veículos. Para aliviar essa situação, sua tarefa como funcionário de uma companhia nacional de táxis é usar técnicas de aprendizagem por reforço para desenvolver uma demonstração de operação dos táxis-inteligentes em tempo real para provar que ambas segurança e eficiência podem ser alcançados.

Arquivos do Projeto

Para essa tarefa, você encontrará o arquivo `smartcab.zip` que contém os arquivos necessários do projeto para download na seção Recursos. *Você também pode visitar nosso GitHub de projetos de Machine Learning para ter acesso a todos os projetos disponíveis para este Nanodegree.*

Avaliação

Seu projeto será avaliado por um revisor da Udacity de acordo com a rubrica de projeto Treine um Táxi-Inteligente para Dirigir. Tenha certeza de ler a rubrica cuidadosamente e autoavaliar-se antes de submeter o projeto. Todos os critérios nessa rubrica devem *atender às especificações* para que você passe.

Arquivos para Submissão

Quando você estiver pronto para submeter seu projeto, reúna os seguintes arquivos e os comprima em um único arquivo para upload. Outra maneira seria fornecer os seguintes arquivos no seu repositório no GitHub em uma pasta nomeada `smartcab` para facilitar o acesso:

- Oarquivo Python `agent.py` com todos os códigos implementados como requisitado nas instruções das tarefas.
- Um relatório do projeto em PDF com o nome report.pdf que responda todas as questões relacionadas as tarefas completadas. Esse arquivo *tem* que estar presente para que seu projeto seja avaliado.

Estou Pronto!

Quando você estiver pronto para submeter seu projeto, clique no botão Enviar Projeto no fim desta página.

Caso você tenha qualquer problema ao submeter seu projeto ou deseje checar o status da sua submissão, por favor, envie um e-mail para ml-suporte@udacity.com ou visite nosso fórum de discussão.

E Agora?

Você receberá um e-mail assim que seu revisor estiver com seu feedback pronto. Enquanto isso, leia seu próximo projeto e sinta-se à vontade para começá-lo ou outros cursos que possam te ajudar!

Note que devido ao conteúdo mais avançado sendo abordado neste projeto, sua submissão pode exigir tempo adicional para ser avaliada. Pedimos a sua compreensão enquanto este projeto está sendo revisado!

Materiais de Apoio

Videos Zip File

Smartcab