1. Comment on your solution to exercise 1

Please see attached `cnn-commented.ipynb` and `cnn-commented.html`.

2. Final Project Topics

Project Ideas are in descending order of preference.

1. Reinforcement Learning: Mario Kart 64 - OpenAI gym

OpenAI provides various game APIs to train agents using reinforcement learning. One of these games is the classic 3D racing game Mario Kart, made available through an N64 emulator. [source](https://github.com/bzier/gym-mupen64plus) Mario Kart is a complex multiplayer game with environmental physics and multiple random power-ups that directly affect gameplay. It is a game of timing, dexterity, and strategy. The game is made more difficult by the need to detect subtle visual stimuli - e.g. the difference between a red shell and a green shell, or the availability of one speed boost vs. three - and determine the correct response to this information, which may vary significantly between outwardly similar situations.

Data/Methods

Because this is a reinforcement learning task, training data can be generated through play, but the fact that this is a multiplayer game makes things more complicated. The team would need to research and adapt the most suitable training paradigms in multiplayer games to solve this task. As in the case of Sokoban, work on the project would initially consist of researching the most promising deep reinforcement learning algorithms. At a later stage of the project, implementation work would begin and 1-3 especially promising algorithms would be compared in their efficacy. (See Algorithms)

The team will probably use techniques like transfer learning from existing networks to simplify the task of identifying visual information in the game, although it is not yet clear how effective this idea will be in practice.

Technical Requirements

Because Mario Kart is played in a complex, 3D, physics-based world, the game can be expected to run relatively slowly. In fact, training time is the expected bottleneck in this project. The team will probably use cloud/server outsourcing, if possible, to decrease training time. While it cannot be predicted exactly, training might take multiple days to produce good results on a standard laptop (in comparison, Q Learning in Bomberman with a 15-dimensional feature space produced reasonable behavior within an hour of training).

One of the main challenges will lie in choosing a neural architecture that accommodates the complexity of the game without weighing down training.

2. Reinforcement Learning: Sokoban - OpenAI gym

Sokoban is Japanese for "warehouse keeper" and a traditional video game. The game is a 2D transportation puzzle, in which the player has to push all boxes in the room onto the storage locations/ targets. The possibility of making irreversible mistakes makes these puzzles so challenging, especially for Reinforcement Learning algorithms, which mostly lack the ability to think ahead. [Paraphrased from source](https://github.com/mpSchrader/gym-sokoban)

Data/Methods

Because this is a reinforcement learning task, training data can be generated through play. Players have access to an approximately 6x6 randomly generated playing field consisting of walkable and unwalkable tiles, target positions, and boxes. Movement is restricted to walkable tiles. The challenge is to develop an agent that can generalize knowledge gained from past board states to effectively solve new, unseen board states. This requires that the agent develop something like "planning ability", since a single misstep can often permanently destroy the agent's ability to solve the puzzle.

Board states can be flipped or rotated to produce new, valid board states (data augmentation). This game would provide an interesting opportunity to compare and evaluate different reinforcement learning techniques on the same task. The team would explore the current literature relating to deep reinforcement learning (or similar state-of-the-art approaches) and evaluate 1-3 of them. (See Algorithms)

Technical Requirements

Because of the simple nature of the game (both abstract gameplay complexity and CPU/GPU/RAM requirements), training can be expected to converge within 24-72 hours on a standard laptop, assuming the learning algorithm is sensibly chosen.

3. Plant Seedlings Classification

The ability to distinguish a weed from a plant is important as it can lead to better crop yields and better environmental protection.

Dataset

The data set (Plant Seedlings data set V2) is publicly accessible at https://vision.eng.au.dk/plant-seedlings-dataset/. It contains about 960 unique plants that belong to 12 different species (Black-grass (334), Charlock (454), Cleavers (348), Common Chickweed (716), Common wheat (258), Fat Hen (543), Loose Silky-bent (805), Maize (260), Scentless

Mayweed (608), Shepherd's Purse (276), Small-flowered Cranesbill (580), Sugar beet (464)) in several growth phases. Further information on the data set can be found at https://arxiv.org/abs/1711.05458.

Methods

Augmentation is used to obtain more training data and to counteract the imbalance of the classes in the labeled data set, by e.g. flipping, mirroring and rotating of images. It is planed to use CNNs to classify the plant species in the image. A classifier will be trained by transfer learning and an own architecture will be created. A comparison of the two is interesting.

Technical Requirements

Minimal computational resources are needed for convergence. The challenge is in choosing a neural architecture to produce *very good* results, as opposed to merely *good* results.

Potential RL Algorithms

The following algorithms and techniques (and potentially others) would be evaluated for their suitability in our reinforcement learning tasks.

- https://ieeexplore.ieee.org/document/8285425
- https://link.springer.com/content/pdf/10.1007%2FBF00992696.pdf
- https://ieeexplore.ieee.org/document/8285400
- https://dl.acm.org/citation.cfm?id=3044850
- https://www.nature.com/articles/nature14236?foxtrotcallback=true
- http://proceedings.mlr.press/v32/silver14.pdf
- https://arxiv.org/abs/1509.02971