1. Start with untrained vanillia neural network
2. Apply small random deviations to network to create a number of similar but unique networks
3. Allow each network to determine strategy for the same race environment/map (the whole race)
4. Use performance to adjust parameters to be better
5. Repeat 2-4 to improve on the population
6. Repeat 1-5 with different initial randomizations and/or network shapes

The network will take in ‘information’ as input and output the optimal velocity for the next time-step

Information:

State:

Car State:

battery voltage, battery/panel temperature

Environment:

Time Dependent:

weather

angle of sun

Distance Dependent:

incline (averaged over a few kilometers)

weather

distance to finish line/ target zone

?? road angle relative to sun

?? traffic

\*\*\* For weather predictions, the further in the future it is, the lower the resolution

\*\*\* For road angle, the further away it is, the lower the resolution, but a roughness factor will be included too

This is for computational speedup

Lots of noise will be added to battery voltage, and perhaps we make a mapping between voltage and charge to reduce error

Noise will also be added to - output velocity, weather (increasing noise with distance in time), Energy usage(skewed towards more usage)

The purpose of noise is to regularize the parameters and make sure that the network is not overfitting - That is, to make sure the success of the network’s decisions is not dependent on following it’s planned strategy perfectly.

By subjecting the model to as many situations as possible, ideally it will learn to generalize the best strategy. Before a race however, it should be trained on maps that are randomly generated to have very similar traits to the race route, or perhaps it could be trained on the race route itself.

The idea is that the network will take in all the relevant information that a person would have and decide the optimal velocity to maintain for the next timestep.

The network doesn’t need to determine the optimal velocity(strategy) for the entire route all at once. Greater uncertainty in weather in the future as well as position of the car may cause errors to add up over time. To re-determine strategy every time information changes unexpectedly is computationally costly and slow.

The disadvantage of the suggested method, is that one cannot know what ‘plan’ it has, as it has no plan and instead is constantly reacting. If an accurate model of the race route could be made, however, it’s possible to simulate what decisions the network would make in a real setting.

Accuracy depends on quality of the simulated model (Could add factor of safety). Probably impractical to train from real vehicle driving, but with gathering data for validation purposes