IDP: Using GAC and IQN based Imitation learning to Construct an Optimal Stochastic Policy for Primate Arm Motion

Gregory Cho, Olivia Langley, Sean Nathan

1 Abstract

This project explores the merits of using generative actor critic (GAC) and implicit quantile network (IQN) based approaches, proposed by Chen Tessler et al. (2019), as an alternative to traditional supervised or imitation learning methods. The objective of these approaches is to have an agent learn an optimal nontrivial stochastic policy in order to ultimately determine a mapping between neural data and arm motion by imitating expert actions, forming an Imitation Distributional Policy (IDP). GAC is an implementation of Distributional Policy Optimization (DPO), which is a solution for continuous control problems with complex non-parametric distributions over continuous action space. Intuitively, as Distributional Policy Optimization learns an internal representation of the environment during training, such as normalized state action values, the GAC learned policy should be more general and thus more applicable to similar environments where the relation between states and actions are not completely analogous to the training set; similar to what is observed with different test subjects for this project's data collection experiments involving simple arm motions. In addition to using a purely GAC based approach, an IQN model is used to measure the efficacy of a simplified supervised method with an assumed Gaussian distribution over action space. While this approach does not incorporate the benefits seen in GAC or the accuracy seen in traditional supervised methods, it is hypothesized that the results are more regularized, minimizing the possibility of overfitting and allowing for the model to be used on other similar state spaces. The code for this project can be found on this public git repository: https://github.com/gwbcho/iqn-imitation-learning.

2 Introduction

As the original generative actor critic (GAC) algorithm is a purely reinforcement learning method, modifications are necessary for it to be usable in an imitation learning setting. This project explores two potential modifications and compares the results of these methods to baseline supervised learning models. The first approach to using GAC as an imitation learning model is to preserve the unmodified agent and, instead, create a custom interactive environment where the rewards are derived from the distance of the generated trajectory to the experts actions. In this approach the model is still engaging in a reinforcement learning task, preserving the internal state action value representations, but is being encouraged towards actions that are considered optimal from an expert agent's perspective. The second modification method is to directly train an IQN model on perturbed expert actions such that the learned probability distribution over actions given the state is equivalent to the distribution of the noise introduced to the expert data. This is more consistent with traditional policy gradient methods with parametric distributions over action space. These two approaches are compared to a purely supervised approach where the mapping of states to actions is seen as a classification problem and solved using both an RNN and simple feed forward neural network. As both approaches used in this project preserve, to some extent a distribution over continuous action space, the end result is an Imitation Distributional Policy (IDP). The hypothesis behind IDP is that, while the

algorithm will likely not succeed in outperforming a pure classification model, it will be better at abstracting an appropriate policy for similar scenarios. Due to limited data, however, this hypothesis is not explored in this project and is, therefore, left as future work.

3 Related Work

Scalable Muscle-Actuated Human Simulation and Control: This paper is the work that is most similar to what we aim to do in this project. They built a comprehensive musculoskeletal model and control system that was able to reproduce human movements driven by the dynamics of muscle contraction. They took into account variations in the anatomic model to account for movements from the highly typical to the highly stylistic. Using deep reinforcement learning, they delve into the scalable and reliable simulation of anatomical features and movements. Their key contribution was using a scalable, two-level imitation learning algorithm. This algorithm was able to deal with the full range of motions in terms of the full-body musculoskeletal model using 346 muscles. They also demonstrated predictive accuracy of motor skills even under varying anatomical conditions ranging from bone deformity, muscle weakness, contracture, and prosthesis use. They also simulate pathological gaits and were able to predictively visualize how orthopedic surgeries would impact the gait of the patients.

Skeleton-Based Action Recognition Using LSTM and CNN: This paper uses a combination of LSTM and CNN in order to conduct effective recognition of skeletal based actions. However, with CNN-based methods, it was found to be inevitable to lose temporal information when the sequence was encoded into images. In order to capture as much temporal information as possible in order to generate an accurate data set for action recognition, the LSTM/CNN combination was used and found to be effective. This paper used the NTU RGB+D dataset for 3D human action analysis. They ended up achieving an 87.4% accuracy rate. In terms of our project, we are using a dataset that takes direct neural signals rather than using spatial and temporal data in order to accurately predict joint actions.

A Neural Network Based Feed Forward Architecture For Recovering 3D Motion Information of Curved Surfaces: This paper is somewhat unrelated to our specific goals, however, they did find ways to recover 3D motion information from 2D flow parameters. Essentially, they were able to discern 3D motions accurately from 2D input data using a network composed of nodes with weightings on node connections to express physical meaning in the 3-dimensional realm when extrapolating the data from 2-dimensional parameters.

4 Methods

As this project aims to use supervised learning to create an effective policy, the original GAC and IQN components mentioned in the original paper, Chen Tessler et al. (2019), require modification before being utilized. Of the numerous modifications possible, this project focuses on testing two approaches which appear to be the most promising. The first approach constructs the set of positive advantage by using a perturbed supervised approach as opposed to using critic and value networks which is what is traditionally used in parametric distributional reinforcement learning. Using this positive advantage set, the network is still able to use the traditional Huber Quantile Loss function to construct a network which emulates a simple parametric target distribution, an approach acceptable for simple tasks but insufficiently flexible to handle a complex distributions over actions. The second approach utilizes the original GAC algorithm in a custom environment which rewards actions similar to the expert agent. This method preserves the internal environment information learned by the model and is able of constructing a more complex distribution over action space by using quantile regression to optimize over action

space as oppose to policy space. Both these approaches are considered imitation learning and are thus labeled Imitation Distributional Policy (IDP) models, with the first method forming a parametric distribution over actions and the second constructing a non parametric distribution.

To assess the efficacy of these modifications to the original algorithm with regard to the task of modeling arm motion from neural data, a baseline comparison is made using a purely supervised model. This model uses direct supervised learning with a standard MSE loss function to train both an RNN and feed forward network on expert data without the need for a target distribution. The loss function would be the difference between predicted actions and expert actions. This approach, while simple, results in a classifier that has no real distribution, negating any of the benefits of using an IQN.

4.1 GAC Training

Intuitively, the GAC algorithm is a means of using quantile regression with a delayed actor critic approach to optimize a distributional policy over policy space as opposed to parameter space (the standard approach for policy gradient methods). For a more detailed explanation of the algorithm, refer to Chen Tessler et al. (2019). GAC theoretically allows for the algorithm to construct complex non-parametric distributions over action space making it more flexible than other baseline RL models.

To capitalize on this model's claimed robustness at solving more complicated tasks, this modification aims to preserve the GAC architecture and provide a custom environment where the agent may learn an optimal policy guided by an expert actor. To accomplish this, the environment rewards actions based on the combined euclidean distance for the state and action vectors.

$$\begin{aligned} R(s, a, s_{expert}, a_{expert}) &= -(D(s, s_{expert}) + D(a, a_{expert})) \\ D(s, s_{expert}) &= \sqrt{(s(1) - s_{expert}(1))^2 + ... + (s(d) - s_{expert}(d))^2} \\ D(a, a_{expert}) &= \sqrt{(a(1) - a_{expert}(1))^2 + ... + (a(c) - a_{expert}(c))^2} \end{aligned} \tag{1}$$

Note that the above equations use d to represent the dimensions of the state space and c to represent the action space. To force the agent to consider the accuracy of past actions, the state space is comprised of concatenated vectors containing previous neural activity and the action the agent took to get to the current state. In this sense the process becomes similar to a Partially Observable Markov Decision Process with the only insight into the true state of the agent provided by the rewards.

The training process for the agent is analogous to the original GAC algorithm from this point on. In order to derive a Monte Carlo approximation of the Huber Quantile Loss between the target distribution, also known as the delayed actor, and the exploratory actor, sampled state, action, and reward tuples are acquired during each training step from a replay buffer which is then used to derive the underlying loss function to conduct back propagation on for the actor network. The critic and value networks are trained in a simple supervised fashion. It should be noted that this process, especially for the AIQN actor, is significantly more computationally expensive than traditional policy gradient methods.

To address the substantial issue of computational cost for this reinforcement learning approach, two different methods are used to limit the time the agent spends on both evaluation and training phases. The first is to limit the maximum steps the agent is able to take before the environment is terminated. Instead of assessing the entire expert sequence, the agent is thus limited to a max step size interval to learn on. While this does limit the accuracy of the model as a whole, it significantly reduces training time. To have the agent learn on all possible state

action pairs, the second method involves using both a limited window size as well as random initialization of the environment after previous terminations where the agent starts from a random expert state and action pair. The results of these attempts to reduce the computational costs of this algorithm are discussed in later sections. To also reduce the necessary computations during training, the size of the action dimension is reduced from 84 values to 27 important ones. This enables the GAC process to more quickly converge to an optimal policy.

4.2 IQN Architecture and Training

Two architectures were used, similar to the GAC process, to test the efficacy of a simplified supervised parametric distributional policy method. The first uses a traditional IQN to determine a parametric distribution where the actions in each dimension of the action vector are assumed to be uncorrelated. The second approach uses an AIQN which assumes that action values are correlated with the previous action. Note that, as both processes use analogous training methods, for the purpose of simplification, both of these network types are considered IQNs.

The architecture for the AIQN network is relatively simple. As with any quantile function, it takes in a random noise vector sampled from any distribution and, to make it pertinent to some state, concatenates said random vector with the state representation. From this, the embedded vector is passed to a GRU which outputs a predicted action along side a hidden state variable, the vector is passed through a single MLP to collapse the embedded vector into a single action element. This process is repeated until the action vector is fully constructed. The GRU component is used to correlate previous actions to the current selection, a process labeled by the authors of the original GAC paper as autoregressive. A graphical representation of this architecture can be seen in Figure 1, note that ϕ is simply some embedding function to map the inputs to the domain $[0, 1]$ which in the paper is set to be a cosine basis linear function.

The architecture of the IQN actor is similar to that of the AIQN. The network still takes as input the current state and a uniform random noise vector. Both vectors are fed into embedding layers which are either MLP's, for states, or a cosine basis linear transformation layer, for noise. From this, the vectors are passed to another set of MLP layers which outputs the entire action vector.

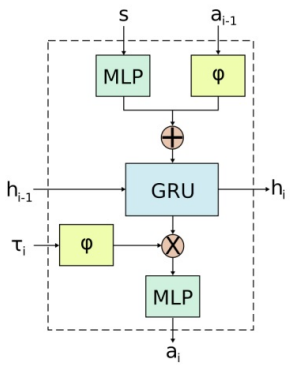

Figure 1: AIQN

The more complicated component is the training mechanism for the network. To ensure that the IQN adheres to some distribution, the training revolves around using the estimated Huber Quantile Loss function of the sampled target distribution of the network. In the original Distributional Policy Optimization algorithm, the sampled distribution is acquired by randomly selecting stored actions from previously observed states. This process is known as memory recall. As there is no sequence of actions in this approach, to replicate this method of Monte Carlo approximation, a batch of expert actions is taken as the target distribution and added to Gaussian noise. This makes the target distribution a Gaussian model with different means per state which must be determined by the network to minimize loss. This process is similar to how parametric policy gradient methods generate optimal stochastic policies.

Once a sufficient sample is acquired to estimate the loss, the network is trained using standard back propagation with a Huber Loss function. This procedure results in a distribution over actions conditional on states. It is clear to see how this algorithm may be used in situations of continuous control. To determine if this algorithm is successful in its task this project aims to determine the total error for each expert trajectory in the test set. This is possible as the observable states in the model (the collected neural activation data) appears independently to the actual action.

4.3 MLP Baseline

To verify that both IQN and GAC approaches are able to perform as well as traditional supervised models, a baseline MLP is used to classify neural states and map said data to actions. This classifier is then used as a comparison to determine how well each model was able to perform given their architecture. The MLP uses the same architecture as the IQN model without the noise embedding layer, with hidden sequential layers of sizes 400, 200, and 200 with the final layer projecting the results to the action dimension. Each hidden layer uses leaky relu as an activation function while the final layer uses tanh.

5 Data and Data Collection

The data is collected using 3D motion capture equipment and an expert agent (i.e. a primate) conducting sever "grasping" tasks. The subject is fitted with up to 120 electrodes which monitor motor cortex activity and is strapped into a contraption which humanely limits non arm related motion. The outputted raw 3D position data is prepossessed by the original experimenters into 32 dimension vectors containing the Euler angles (relative position metrics) of each joint. In order to construct a sufficient reinforcement learning agent from this information, the algorithm under development would have to translate the sequences of Euler angle data, effectively a hidden state representation in the underlying MDP, to some expert associated action space. To do this, this project takes the approach of simply taking the difference between two recorded Euler angle vectors within the sequence and labeling that as the adjustment vector, which may be considered, crudely, as the expert action.

6 Efforts and Challenges

The primary challenge to implementing this approach to imitation learning involves the IDP algorithm's computational cost. While the non RNN related approaches, such as the IQN or FFN, trained in reasonable amounts of time (usually no more than 1-30 hours), the RNN approaches such as the AIQN and RNN methods required substantial amounts of time to train, often taking 4 times longer on an Nvidia Tesla P100. This is a substantial limiting factor to this algorithm's applicability to real world problems and limited the number of experiments IDP was able to run. As the AIQN and RNN methods were exceedingly expensive, tests using them were either terminated after a limited number of training iterations, or in the case of the IDP and baseline methods, not tested at all. Another issue encountered was the limited data available for model training and evaluation. The primary reason why IDP could not be assessed against contemporary methods for different subjects performing similar tasks was that there was insufficient processed data to perform such an experiment. As this is the case, this report instead focused on proving that the algorithms proposed could learn an imitation learning policy in a fashion similar to that of a traditional network.

In addition to the technical limitations there were several other issues encountered with the GCP VM instances used. While running one experiment, the results file was overridden after the completion of the program and a new experiment was initiated without user input. This led to a significant loss of valuable data for the GAC IQN method, which in reality achieved rewards closer to -3000 as opposed to the -5000 indicated, forcing the graph in the results section to use incomplete snapshot data.

7 Results

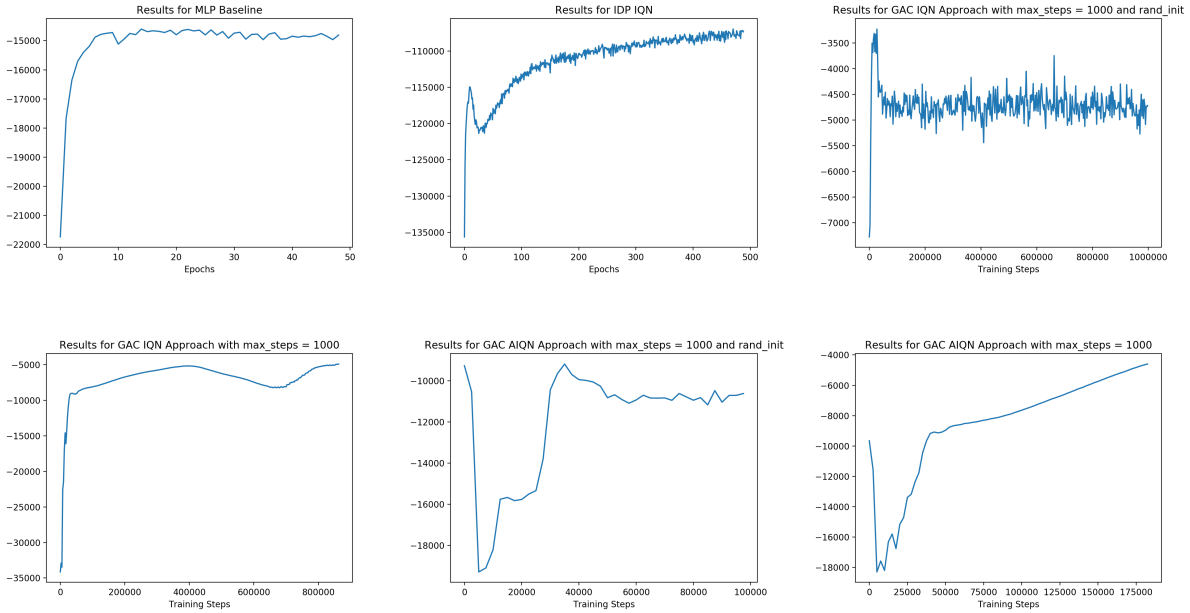

Figure 2: The top row graphs show results from the baseline MLP classifier (top left), the simplified IDP IQN method (top middle) trained over the entire expert data set, and the GAC (IQN) approach (top right) with max steps set to 1000 as well as random initialization per every environment reset. The bottom row shows the results from the GAC (IQN) approach without random initialization (bottom left), GAC (AIQN) with random initialization (bottom middle), and GAC (AIQN) without random initialization (bottom right). For the IDP IQN and MLP methods, the x axis represents the number of epochs and the y axis represents the negative cumulative distance from the expert achieved by the resulting trajectory. In addition, for these supervised methods, the training set consisted of 90% of the expert data with the validation set containing the remainder. For the GAC methods, the x axis represents the number of training steps and the y axis represents the cumulative rewards achieved by the agent. For all GAC methods the expert training environment was terminated after 1000 steps. Due to limitations in time and computational ability, most training operations were terminated after it reached a point where no significant changes were observed.

The IDP GAC and IQN methods were compared to a baseline MLP model to measure their efficacy at predicting arm motions from neural data. This process was selected as a means of evaluation as the algorithm's objective isn't to perfectly replicate the expert but, instead, to determine an optimal stochastic policy enabling expert like actions given neural activity as a state representation. Since a standard MLP classifier is the most obvious solution to the problem of determining a function approximation to represent state action mappings, to show that IDP performs as well as an MLP in this task, bearing in mind that this is merely a proof of concept attempt, would appear to be sufficient empirical substantiation of it's efficacy; demonstrating that IDP methods are at least an alternative if not superior approach. To quantify the results of the experiments conducted, for the purposes of this report, any per action value difference less than or equal to 0.2 degrees from the expert was considered sufficiently similar to be an alternative approach. This places a cap on the error available for the stochastic methods being compared in this report.

The results indicate that most IDP GAC methods are at least as good as the MLP baseline, diverging only slightly from expert actions per step (see Figure 3). The GAC IQN method

Method	Max	Avg Dist Per Step	Avg Dist Per Value
MLP Baseline Classifier	−14599.78	1.69	0.06
IDP IQN	−111736.50	12.98	0.48
GAC (IQN + rand init)	−4602.11	2.30	0.08
GAC (IQN)	−4889.75	2.40	0.09
GAC (AIQN + rand init)	−9184.69	4.59	0.17
GAC (AIQN)	−4597.58	2.30	0.08

Figure 3: The max reward achieved, average distance per step values, and average distance per action value for each method. Note that the total number of steps varies per method. For the entire expert trajectory there are 86517 states and actions. IQN GAC methods only train on 1000 sequential steps of the entire expert set while baseline and IDP IQN methods train on the entire training set. Note that the average distance per step metric measures the euclidean distance per action vector in which there are 27 action values. This value is approximated by taking the max rewards achieved, dividing the result by 2 if the process is a GAC method to account for the over-counting that occurs during training, and dividing the result by the number of evaluation steps, which, for the purposes of this experiment, are set to 1000 for GAC and 8651 for the IDP as well as MLP supervised methods.

without random initialization appears to perform similarly to the baseline approach, achieving an average distance per step of 2.5 as opposed to 1.71. The average per action value distance for this method was 0.09 (derived by taking the average distance per step and dividing it by the action dimension) while the baseline achieved a value of 0.06, meaning that each joint was on average only off by 0.09 degrees, indicating expert like behavior. Similar results are seen from the GAC AIQN approach without random initialization. The method's average per action value distance was 0.08 and it's average per action error was only slightly above the baseline. While not as pronounced as the results from the runs without random initialization, the corresponding GAC attempts that took a Monte Carlo approach to exploring the entire state space did also achieve relatively consistent performances when compared to the MLP classifier. It is likely that the slightly poorer performance seen in the IDP GAC methods is a result of the stochastic nature by which actions are selected. The next steps would be to assess the efficacy of these models on different subjects performing similar actions and compare their performance to the baseline.

Note that the IDP IQN approach failed to achieve sufficient end rewards and per step distances to be considered suitable replacements for the baseline method. It is possible that this was due to the noise present in the model for the process. While this method did not surpass the baseline network performance, it is still possible that it could outperform the MLP approach for neural data collected from different subjects.

Additional noteworthy observations from the data are the relatively slow rates of learning for the GAC methods and the prominent spike in training rewards for the IDP IQN approach in it's early epochs. The slow training rates appear to be a result of the environment used for the GAC reinforcement learning task. Each reward per action is determined by the correctness of the previous action resulting in a target policy which mimics the expert agent. As the agent proceeds through the environment, though, the information as to the correctness of the current state is lost and the entire expert sequence must play out until another attempt can occur. This potentially leads to a loss of valuable information, increasing the amount of time necessary till training determines the optimal policy. To address this issue it may be effective to increase the memory of the state, outputting rewards that better reflect the correctness of the arm position given all previous actions. This would drastically increase the state space which, in turn, would require the algorithm to utilize more expert data to generate a more complete model. It is currently unknown as to why the training curve spikes initially for IDP IQN. It is possible that

it is initially overfitting the data before discovering, through exploration, a newer more optimal policy.

8 Discussion

The results of these experiments are, on the surface, promising, however, there are still concerns with the performance of the GAC based approaches using random initialization. While their per action value error is relatively low (0.22 for the IQN approach and 0.19 for the AIQN approach) their training behavior is inconsistent with expectations. The training curve appears to minimally change after a certain number of training runs, appearing to find a local optimal as opposed to the global optimal policy. As such, while the error may be relatively minimal, the actual motion of the arm may not represent the actions of the expert sufficiently well. Further assessment is necessary to determine if using random initialization is an appropriate means of rewards shaping based imitation learning. Visualizing arm motion would assist in human experimenters determining the success of failure of GAC as an imitation learning method. A better loss function and more appropriate reward shaping than the those currently used in this project would also be useful in better assessing the efficacy of IDP methods.

9 Future Work

While this project shows that IDP methods can learn expert policies sufficiently well, it still remains to be seen whether this form of imitation learning is more applicable to non analogous environments, such as different subjects with different neural data performing similar tasks. As this is the entire reason for this approach, using data from different subjects performing the same task as a validation set would, without significant modification to the code, greatly increase the value of this experiment. In addition, further investigation is necessary to evaluate the training behavior of the IDP methods. At the moment all explanations provided are effectively speculation as, due to time constraints, no studies were conducted to substantiate the claims made. It would be interesting to more formally determine the dynamics of the created environment as well as the model itself.

Future work could also focus on applying modifications that would improve the overall quality of the reinforcement learning models used. In particular, the environment used for the GAC methods does not fully adhere to an MDP format which best captures the objective of the agent. To ensure the model can more accurately imitate expert actions, instead of using a single previous action to determine the next reward, it would appear reasonable to use all past actions to do so. The only limiting factor to this would be space complexity. As each state would now contain the whole trajectory up till that point, for long sequences of actions the space required to store the necessary state vector would balloon considerably.

10 Ethical Considerations

The data set we are utilizing for this project was collected from primate subjects using a combination of marker-based motion capture and cortical recordings. While the use of motion capture and markers is relatively benign, the neural recording process relies on the implantation of microelectrode arrays, which is an invasive procedure. Additionally, the primates underwent a training process to intercept objects being swung toward them, thus eliciting the necessary reach-to-grasp motion being recorded. This collection process brings up the question of animal treatment in research, and the ethics of performing invasive procedures or intensive training processes on a test subject which cannot consent to this research.

As the results of this research are meant to be applied to the development of neuroprosthetic limbs, the "stakeholders" in this problem are humans who rely on prosthetic devices. This means that any mistakes in the algorithm could have significant consequences for anyone who uses a prosthetic system developed from this research. Ideally a small mistake would only result in decreased accuracy of the model, but a mistake resulting in the miscorrelation of neural activity and joint movement could potentially be dangerous in a real-world scenario. That being said, the likelihood of these types of algorithmic mistakes being propagated into the final neuroprosthetic technology would hopefully be mitigated by the comprehensive additional testing required for these devices.

Energy consumption is another ethical consideration with regards to this technology. Each method took significantly longer to train than traditional RNN or policy gradient algorithms. The GAC approaches took up to 2 days for IQNs and 8 for AIQNs. These algorithms are, thus, substantially energy intensive leading to this report's apprehension to endorse such approaches for simple classification problems or imitation learning, regardless of their potential benefits.

11 Conclusion

This project successfully demonstrates that reward shaping to construct an imitation learning method from the GAC RL algorithm leads to promising results which even outperform a baseline classifier when compared to the same data set. The project results also show that the simplified application of IQNs through the use of IDP methods leads to sub par network performance, which was expected. However, the findings of these experiments do not provide additional pertinent information regarding the efficacy of these methods when used on non analogous data sets. Until experiments can be conducted where such a comparison can be made there is very little to discern from this report's findings other than that the original hypothesis proposed, that IDPs can perform as well if not better than a baseline MLP supervised model, is still plausible.

While the outcomes of this project's experiments were promising, there are still several major drawbacks to some of IDP's methods that may limit its practical applications. These issues need to be addressed or better understood before considering this technology for deployment in an industry or rigorous academic setting. The primary issue lies with the AIQN approach as the duration of training necessary till it achieves optimal results is often significantly longer than other methods. Despite the fact that the number of training iterations remain similar to that of a traditional IQN, the AIQN often takes over 4 times longer to complete. This computational burden imposed by the network makes its value as a replacement for faster, less accurate, approaches, such as IQN or MLPs, dubious. Furthermore, the algorithms that performed best in this project's experiments were also the most data hungry. The GAC approach required the concatenation of neural activation data and prior actions, resulting in a state space that was nk in size (let n be the original action space size and let k be the size of the state space). In order to construct a more complete model, the GAC approach would require expert actions for a large subset of this state space, leading to the issue of data collection and sparsity which are common issues in imitation learning problems.

12 References

1. Tessler, Chen; Tennenholtz, Guy; Mannor, Shie. "Distributional Policy Optimization: An Alternative Approach for Continuous Control". https://arxiv.org/pdf/1905.09855.pdf

2. Roger Koenker; Kevin Hallock. "Quantile regression: An introduction". Journal of Economic Perspectives, 15(4):43–56, 2001.

3. Roger Koenker; Zhijie Xiao. "Quantile autoregression". Journal of the American Statistical Association, 101(475):980–990, 2006.

13 Appendix

13.1 GAC IQN Performance on a Validation Set

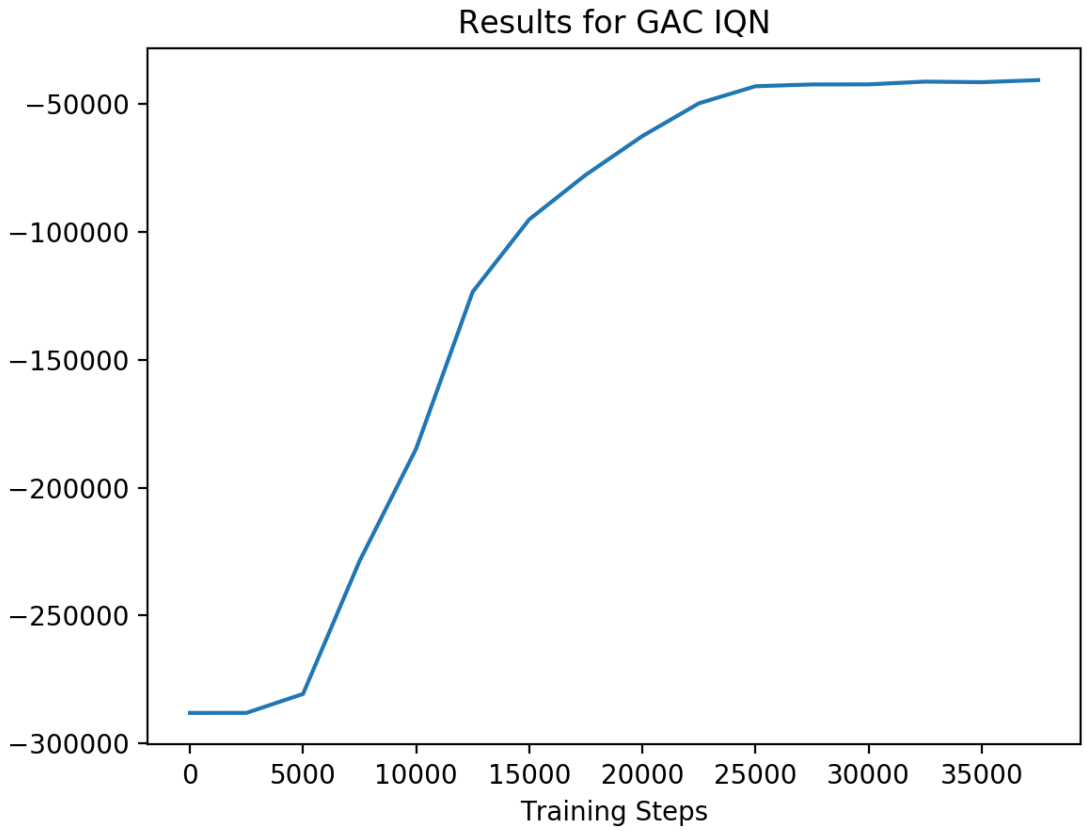

Figure 4: The results of GAC IQN on a validation set. The x axis represents training steps and the y axis represents cumulative rewards. The maximum reward achieved was −40788.29 which results in an average per action distance of 2.35 and a per value distance of 0.09, consistent with the results of the previous experiments. As with the supervised validation methods, the training set consists of 90% of the data while the validation set comprises the remaining 10%.

The GAC IQN method, when tested on a validation set, appeared to perform as well as it did in the original experiments. While this is not salient to the objective of this report it is still information of value and makes the claim of its potential use as an alternative to traditional classifier methods significantly stronger.

13.2 GAC AIQN Performance on a Validation Set

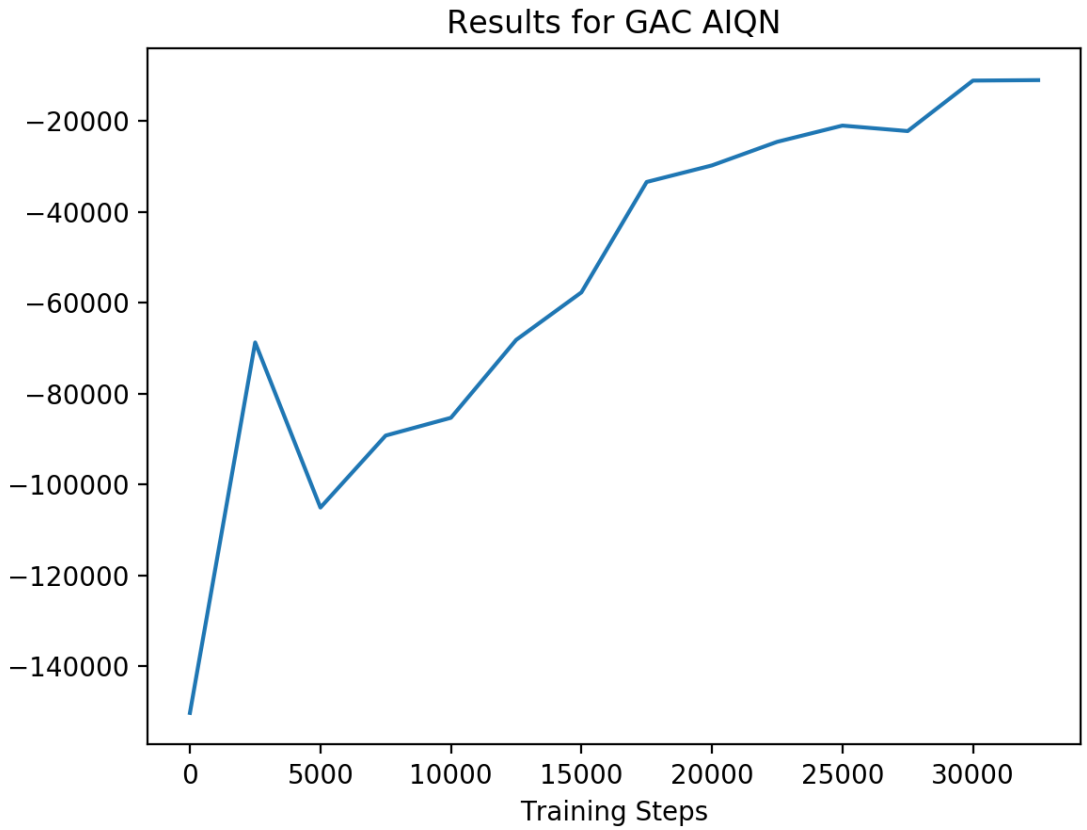

Figure 5: The results of GAC AIQN on a validation set. The x axis represents training steps and the y axis represents cumulative rewards. The maximum reward achieved was −11067.95 which results in an average per action distance of 0.63 and a per value distance of 0.02, better than the results of the previous experiments. As with the supervised validation methods, the training set consists of 90% of the data while the validation set comprises the remaining 10%.

The GAC AIQN method, when tested on a validation set, appeared to perform even better than it did in the original experiments even surpassing the performance of the baseline MLP classifier. While this, like the previous observations made on GAC IQN's performance on a validation set, is not salient to the objective of this report it is still information of value and makes the claim of this algorithm's potential use as an alternative to traditional classifier methods significantly stronger.