Imitation Learning with Inconsistent Demonstrations through Uncertainty-based Data Manipulation

Valletta, Peter; Pérez-Dattari, Rodrigo; Kober, Jens

DOI
10.1109/ICRA48506.2021.9561686

Publication date
2021

Document Version
Final published version

Published in
Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2021

2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

2021 IEEE International Conference on Robotics and Automation (ICRA) | 978-1-7281-9077-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICRA48506.2021.9561686

Imitation Learning with Inconsistent Demonstrations through Uncertainty-based Data Manipulation

Peter Valletta[1], Rodrigo Pérez-Dattari[1] and Jens Kober[1]

Abstract— Aleatoric uncertainty estimation, based on the observed training data, is applied for the detection of conflicts in a demonstration data set. The particular focus of this paper is the resolution of conflicting data resulting from scenarios with equivalent action choices, such as obstacle avoidance, path planning or multiple joint configurations. In terms of the estimated uncertainty, the proposed algorithm aims to decrease this otherwise irreducible value through direct alteration of the accrued data set and to provide data that a policy-learning neural network is able to fit appropriately. The proposed algorithm was validated with real robot scenarios while learning from inconsistent demonstrations, where the resulting policies consistently achieved their prescribed objectives. A video showing our method and experiments can be found at: https://youtu.be/oGYnzlW9Ncw.

I. Introduction

Imitation Learning (IL) offers an intuitive methodology for humans to teach robots a set of motions, without demanding them to be skilled in robotics. This is especially beneficial as these tasks become increasingly complex and manual prescription of such behaviors becomes a daunting proposition. IL builds on the premise that it is frequently easier for a human teacher to demonstrate the sought behavior, rather than explicitly engineer it, and incorporates human knowledge into the learning process, significantly decreasing sample complexity [1]. A comprehensive approach to devising IL strategies is encompassed in four key questions - when, what, who and how to imitate a teacher's input [2], [3].

These questions are rigorously addressed in literature, therefore this work is concerned with alternative issues that persist as open problems within the same field [1]. Namely, this work addresses the frequent assumption within IL methodologies that user input is ideal and consistently leads to optimal task learning. A realistic set of demonstrations will contain conflicting or poor examples that degrade the quality of the learned policies and could lead to failure in meeting the required goals, notwithstanding a significant amount of data. In order to resolve this matter of poorly learned policies despite the accrual of a comprehensive amount of demonstration data, we present a framework that exploits Neural Network (NN) based uncertainty estimation (UE) to detect conflicting regions in our data set and automatically resolve these conflicts, in settings with equivalent action choices. An adept example of this setting is the obstacle avoidance scenario, where a moving agent must avoid an obstacle by moving either left or right, with either action being equally valid, as shown in Figure 1. Path planning and robotic platforms with multiple joint configurations are other scenarios which present similar issues.

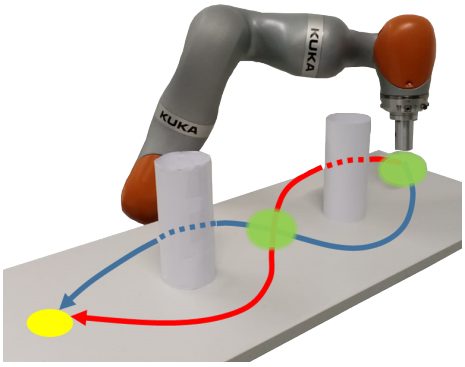


Fig. 1: An obstacle avoidance scenario with two obstacles, applied as an experimental setting in this paper. The yellow circle represents the goal position for the arm's end effector, while the green circles denote areas where multiple paths may be chosen while traversing this environment.

This paper explores the use of aleatoric UE to detect conflicts in IL training data and aims to reduce the predominant sources of this uncertainty through modification of the available data. The assumption of equivalent action choices dictates that every action is equally valid to solve a task, which allows for the resolution of detected conflicts via the removal of data points. After identifying conflicting elements within a data set, these are grouped into one or more clusters of closely-located points. An action choice is selected from every cluster, based on its frequency, and all data within the relevant cluster belonging to other action choices is removed from the data set, thus resolving the conflict. The adjusted data set is applied to update the policy being learned. Notably, the algorithm proposed is composed of multiple, distinct elements, which may be varied, modified or improved upon to suit different use cases and implementations, particularly as this method is applied to more complex scenarios.

This paper continues as follows: Chapter II presents the literature most closely related to this work and briefly discusses its main theoretical aspects. Chapter III outlines our contributions and Chapter IV presents the results obtained from multiple experiments. Finally, our concluding remarks are included in Chapter V.

[1]All authors are with the Cognitive Robotics Department, TU Delft, Delft, The Netherlands (valletta.p@gmail.com, {r.j.perezdattari, j.kober}@tudelft.nl)

II. Related Work

A. Uncertainty Estimation

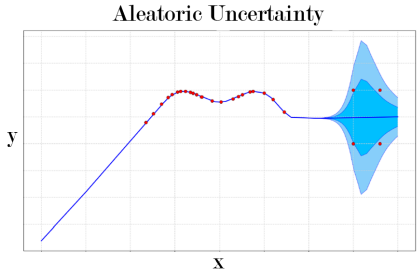
A general use of the term uncertainty refers to all the situations where there is a lack of knowledge with regards to the outcome of a problem, its structure or the nature of the factors motivating the problem. Uncertainty is categorized as being either epistemic or aleatoric, with their amalgamation defined as predictive uncertainty, encapsulating a model's certainty (or lack of) in its output [4]. The former is briefly introduced for the sake of theoretical rigor but is not a concern of this work, which focuses on aleatoric UE with NNs, which is an active field of research [5]–[10]. Figure 2 presents a visual comparison of aleatoric and epistemic uncertainty.

Epistemic uncertainty, also known as model uncertainty, accounts for doubt in model parameters, structure or a general lack of knowledge about the problem at hand and may be nullified through additional information. A simple approach to quantifying this uncertainty involves computing the variance between a set of outputs obtained from an ensemble of deterministic models, such as Neural Networks [11], however the produced value may not always be accurate and this approach has been criticised for not performing Bayesian inference [4], [12]. Anchored ensembles of Neural Networks [12] and the application of dropout in a work's testing phase [4] have been shown to approximate Bayesian inference and thus adequately estimate this uncertainty.

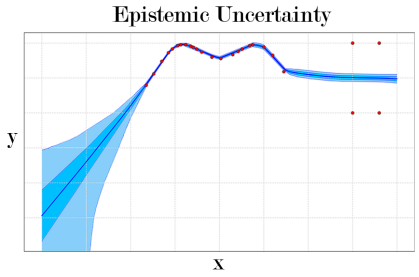
Aleatoric uncertainty, also known as data uncertainty, is a measure of the network's confidence in the accuracy of its output, with any doubt stemming from inherent randomness within the environment or the agent. This paper exclusively considers data uncertainty in the analysis of Imitation Learning (IL) demonstration data, given the objective set out in Section I. An accurate estimate of this uncertainty may be efficiently obtained while learning a regression task with a feed-forward Neural Network through supervised learning (SL), with the aim of maximising the log likelihood of the target data, in both uni- and multi-dimensional scenarios [4], [13]. This approach, which is especially relevant to this work, requires the addition of a secondary network output to predict the uncertainty alongside the learned quantity. Other network configurations estimating aleatoric uncertainty with neural networks are observed in [6], [14].

B. Learning from Conflicting Data

A family of methods implicitly address the challenge of learning from conflicting data by minimizing the reverse Kullback-Leibler (KL) divergence, or I-projection, which collapses the output of the learned models to one of the demonstrated modes [15]–[17]. Even though these approaches are promising, the minimization of the reverse KL divergence has shown to be a challenging problem and, therefore, it is an active field of research. Contrary to our work, these methods are not uncertainty aware and data manipulation techniques are not applied.



(a) Aleatoric uncertainty estimation.



(b) Epistemic uncertainty estimation.

Fig. 2: One-dimensional example of a comparison between estimated aleatoric and epistemic uncertainty over a data set [4], [12]. In regions with multiple values of y for the same x, aleatoric uncertainty increases. In regions without data, epistemic uncertainty increases.

The identification and resolution of conflicting data, which is the objective of this paper, is not an extensively explored area within the IL field. Ledesma et al. [18] propose a deterministic method for the computation of a measure of the conflict between data points within data sets compiled for supervised learning; however, this is not linked with a methodology for the removal or correction of this data.

Chernova et al. [19] introduce a methodology for the successive detection and resolution of data conflicts arising from situations involving equivalent action choices. The algorithm leverages classification confidence predictions obtained from a Gaussian Mixture Model (GMM) classifier to identify regions of low confidence points, which are subsequently gathered into distinct clusters. Differing data points within unique clusters are transformed to a single value, selected from a finite set of options within each cluster.

The method proposed in this work is related to the GMM-based, conflict resolution approach proposed in [19]; however, it extends their framework to the continuous action domain, which is the prevalent scenario in robotic problems. Furthermore, neural networks are applied within this paper due to their scalability properties, thus removing the computational limitations of a GMM-based implementation.

III. Methodology

This section describes the proposed data manipulation framework. The succinct view is as follows: (1) Aleatoric uncertainty is estimated for every data point in the demonstration data set using a NN, (2) Clustering is

applied to detect different regions of uncertain points, (3) Action choices are estimated for every cluster, (4) One action choice is selected for every cluster; all other options are removed, and (5) the NN is retrained using the modified database.

A. Aleatoric Uncertainty Estimation

Aleatoric uncertainty is estimated with a methodology that trains a NN in a supervised fashion to predict the mean and variance of a target probability distribution [13], where the predicted variance is interpreted as the uncertainty about that mean value. The estimation of uncertainty is an effective way of identifying one or more regions with potentially conflicting data in the data set and is a logical first step that allows for more directed methods further down the line.

The method applied in this work assumes that for an input $\boldsymbol{x} \in \mathcal{X}$, the collected data contains data-dependent additive white Gaussian noise $n(\boldsymbol{x})$; individual points are defined as

$$d(\boldsymbol{x}) = f(\boldsymbol{x}) + n(\boldsymbol{x}), \tag{1}$$

where $f(\boldsymbol{x})$ is the true function generating said data. The definition of observation noise as a function of the data itself implies that it may be learned by the network as the covariance about $f(\boldsymbol{x})$ [8]. This covariance is interpreted as the uncertainty regarding the mean prediction of $f(\boldsymbol{x})$, both of which are learned concurrently. The estimation of two values specifies the addition of output neurons and hidden layers to form a split-network architecture with two distinct groups of output neurons corresponding to each quantity to estimate, linked to one set of inputs. From the normal distribution assumption for $n(\boldsymbol{x})$, and setting the network's goal as the maximization of the log-likelihood of the target values $\boldsymbol{y} \in \mathcal{Y}$, it follows that its loss function is defined as

$$\mathcal{L} = \frac{1}{2}\left[\boldsymbol{y} - f(\boldsymbol{x})\right]^T \upsilon(\boldsymbol{x})^{-1}\left[\boldsymbol{y} - f(\boldsymbol{x})\right] + \frac{1}{2}\ln\left|\upsilon(\boldsymbol{x})\right| \tag{2}$$

in the multi-dimensional setting, where the mean prediction $f(\boldsymbol{x})$ and covariance prediction $\upsilon(\boldsymbol{x})$ are commonly trained simultaneously [4], [13]. In this case study, $\mathcal{X}$ corresponds to the state space of the robot and $\mathcal{Y}$ to its action space.

The computation of the covariance matrix presents two major challenges:

1) As the number of dimensions in the action space increases, the number of dimensions of the covariance matrix increases quadratically, which may become intractable as the action space grows.
2) The covariance matrix must be positive semi-definite. As a consequence, a strategy has to be adopted in order to enforce this property at the output of the NN.

To address these limitations, when estimating the uncertainty of our system, the dimensions of the action space are assumed to be uncorrelated - consequently, it is only necessary to estimate the main diagonal of the covariance matrix, i.e., the problem is reduced to the estimation of $\mathcal{N}$ independent one-dimensional normal distributions, where $\mathcal{N}$ corresponds to the number of dimensions in the action space. Then, the number of values to estimate in the covariance matrix increases linearly with the number of action dimensions. Finally, to enforce the covariance matrix to be positive semi-definite, every value of the diagonal must be equal to or greater than zero. This is achieved by using the softplus activation function.

Until now, we have made two strong assumptions regarding our data: 1) the target probability distribution follows a normal distribution, and 2) the dimensions of this distribution are uncorrelated. In our case-study (equivalent action choices) the former does not hold, since equivalent action settings imply that the target distribution is multi-modal, which is not the case for a normal distribution. Likewise, the latter assumption only holds if the orientation of each of the variances aligns with the corresponding axes in the action space. Nevertheless, the current objective at this stage in the framework is to obtain a scalar value that represents the uncertainty in our data, i.e., for every input in our explored state space (negligible model uncertainty), it is desired to know how certain or uncertain the NN is about its output. A normal distribution with uncorrelated dimensions suffices to capture this uncertainty, thus the failings of these assumptions are overlooked and the Frobenius norm of the covariance matrix is computed to combine the uncertainty of every dimension into a single scalar value.

One final remark; in order to accelerate and improve the training stability of our algorithm, the training of the mean and uncertainty prediction halves of the network were separated into two distinct processes, similarly to the methodology proposed in [13]. First, the mean prediction half is trained exclusively with a mean squared error (MSE) loss, until the network accurately represents the training data. The relevant network parameters are subsequently frozen and the remaining, covariance-related branch of the network is trained with the loss function (2). Note that this sequential training is equivalent to training both outputs together using (2), since in both cases $f(\boldsymbol{x})$ will converge to the mean of the samples.

B. Data Conflict Resolution

Our algorithm is equally effective with any iterative and non-iterative Imitation Learning (IL) approaches that gather demonstration data in a database and can process both continuous and discrete action spaces. The case of continuous actions is the focus of this paper, given its greater generality and prevalence in the robotics field. In the ensuing stages of the proposed method, the assumption regarding the correlation between the dimensions in the action space is withdrawn, ensuring the generality of this approach.

1) Inconsistencies detection: Following the collection of demonstration data and the evaluation of uncertainty within said data, highly uncertain data points are selected based on a user-defined threshold, τ. The application of a user-defined threshold, rather than a homogeneous approach such as the mean or median uncertainty value, was found to be the most effective in isolating highly uncertain data points within the data set. Such a collection of points does not provide information with regards to their distribution or relationships within the state space, thus an algorithm was required to define groups (events) of closely-located, highly-uncertain data, which were most likely to be conflicting and thus, were regarded as candidates for review and correction.

2) Clustering of inconsistent events: Density-based Spatial Clustering of Applications with Noise (DBSCAN) [20] was introduced within this approach to achieve this objective, due to its high efficiency, ability to identify arbitrarily-shaped clusters and a limited dependence on prior knowledge. A cluster is defined when at least 3 points lie within a region with radius $\varepsilon \cdot m$ from a given point. The variable ε denotes the average distance between data points in the complete space, while m is a user-defined multiplication factor. The use of the parameter m allowed for a more accurate definition of these clusters and the inclusion of all the relevant points. Alternative clustering methods may replace DBSCAN, depending on the scenario being considered.

3) Action choice selection: In subsequence to the clustering step, the developed framework contained one or more groups of data, each possibly containing multiple action choices leading to data conflicts. Data conflicts were addressed through the selection of one action choice per cluster; data points belonging to the remaining action choices are removed from the data set. Given that we are working with continuous actions, an action choice must be defined as a region, not a single value. To find such regions, the incidence of actions within a cluster must be computed. An efficient approach is Kernel Density Estimation (KDE) [21], a non-parametric method for probability density modeling which approximates a probability density function $p(\boldsymbol{x})$ from a set of observations which are assumed to be drawn from the same function. A Gaussian kernel was chosen to approximate sampled distributions.

After fitting a density model in the action space of each cluster, we would expect to obtain multi-modal distributions, where each modality would correspond to one of the action choices. Therefore, our method selects one of the action choices by defining a region around the action value a^s with the highest density. Every data point with an action value a below a minimum distance r to a^s belongs to this region. Otherwise, it is considered as a different action choice and it is removed from the data set. It is worth mentioning that this approach gives preference to the most frequent action choices observed in the demonstrations; however, given our assumption of equivalent action choices, any heuristic that selects one modality per cluster would work.

Algorithm 1: The Uncertainty Manipulation algorithm

```
procedure
    given D, τ, m, r
    υ ← Estimate uncertainty in D
    M ← data with υ > τ in D
    ε ← mean nearest neighbour distance from D
    C ← set of clusters in M with DBSCAN(ε · m)
    for all c ∈ C do
        n ← number of different actions in c
        if size(c) ≥ 3 and n > 1 then
            a^s ← Most frequent action in c
            for all a ∈ c do
                if ||a − a^s|| > r then
                    D_new ← remove a from D
    Update policy on D_new
```

Finally, the NN is trained once again using the modified database. A pseudocode representation of the proposed algorithm is included in Algorithm 1, where $\mathcal{D}$ denotes the gathered set of demonstration data. In this algorithm, $M \in \mathcal{D}$ is a subset of highly uncertain data within $\mathcal{D}$, while $C \in M$ is a set of clustered data points grouped together by the DBSCAN algorithm.

IV. Experimental Results

Two experiments were carried out in the real world using a KUKA LBR iiwa 7 manipulator. The robot was able to move in the XY plane, parallel to a flat table with various obstacles on it, as shown in Figures 1 and 3. The velocity of the robot was controlled in task space with fixed orientation. The state of the robot was represented by its (x,y) position. As a consequence, both the state and action space were 2-dimensional. Demonstrations were obtained by teleoperating the robot directly in task space using a space mouse; consequently, it was possible to directly imitate this behavior with NNs and behavioral cloning. The same algorithm hyper-parameters were used for both experiments: $\tau = 0.4$, $m = 6$, $r = 0.1$ and a bandwidth of 0.15 for the KDE. The hyper-parameter tuning was a straightforward process, given that every step of our algorithm can be visualized in plots (as it can be observed in the following subsections) and the values of the hyper-parameters can be directly inferred from them. However, as the dimensionality of the state space grows, this approach can become intractable and other techniques could be required to find these values.

A. Scenario 1: One obstacle

As seen in Figure 3, the objective of this experiment was for the manipulator to travel from one side of an obstacle to another while avoiding collisions. Ten demonstrations were collected. Given that it is possible to achieve the goal of the task by choosing to avoid the box from one side or the other, conflicting samples

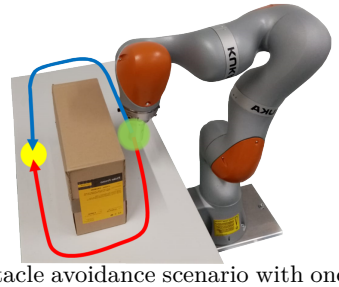


Fig. 3: An obstacle avoidance scenario with one obstacle. The yellow circle represents the goal position for the arm's end-effector, while the green circles denote areas where multiple paths may be chosen while traversing this environment.

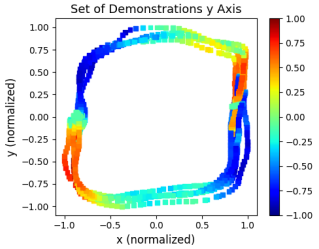


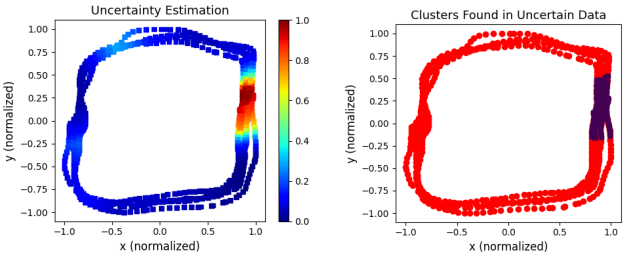
Fig. 4: Demonstrated trajectories in the XY plane. The color intensity of each point indicates the value of the demonstrated action in the y axis.

were obtained in the region that covers the starting positions of the arm's end-effector. This can be observed in Figure 4, where around the coordinates (0.9, 0.0) conflicts in the action dimension along the y axis are observed, which corresponds to the green region in Figure 3. The goal is located around the point (-0.8, 0.0).

Figure 5 depicts that, as expected, this region presents high uncertainty and that one cluster (in purple) was detected by the clustering algorithm, DBSCAN.

Within the detected cluster, Figure 6 exhibits the bimodal distribution obtained in the action space using KDE. Two peaks were found around the points (-0.6, -0.6) and (-0.6, 0.6) (the lighter the value, the higher its probability).

Finally, in Figure 7 it can be observed that the value at the coordinate (-0.6, 0.6) in Figure 6 was chosen by our method to modify the demonstrations and to retrain the network. As a consequence of modifying this set of action values, the aleatoric uncertainty was removed from this



(a) Estimated aleatoric uncertainty of the data. The color of the data indicates the uncertainty computed at each point.

(b) Points in red represent the complete database and points in purple belong to the detected cluster.

Fig. 5: Clustering of inconsistent events.

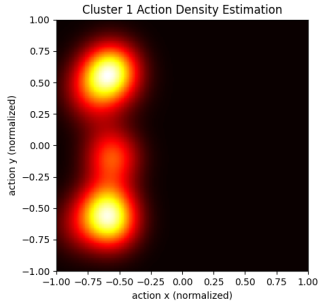
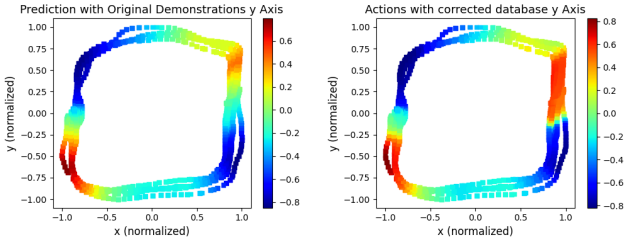


Fig. 6: Density estimation in the action space of cluster 1.



(a) Neural Network regression using original demonstrations.

(b) Neural Network regression using modified demonstrations.

Fig. 7: Comparison of Neural Network regression before and after modifying the demonstrations.

region.

Both policies depicted in Figure 7 were deployed in the robot for twenty evaluations each (with different initial points) and the results compared. The policy trained with the original demonstrations was able to successfully solve the task 60% of the time, while the policy trained over the modified demonstrations was successful in every trial. A trial was considered to be successful if the manipulator was able to reach the goal without colliding with the obstacle. The failures observed in the policy trained with the original demonstrations were due to the fact that in the conflicting region, the policy was computing an average of the demonstrations, which lead to slow or no movement at all. This behavior also made the arm occasionally diverge away from the demonstrations and thus failing to complete the task.

B. Scenario 2: Two obstacles

This scenario is depicted in Figure 1. The objective is to reach the goal position, shown in yellow, while avoiding both obstacles. In this case, two regions of uncertainty are generated by the demonstrations, one at the beginning of the motion and the other centrally, between both obstacles. Due to the increased velocity of the manipulator as it moves towards the goal, which yielded a higher risk of collision with the second obstacle, this scenario proved to be more challenging than the preceding one. Eight demonstrations were carried out.

The inconsistencies presented in the demonstrations can be observed in Figure 8. As opposed to the previous scenario, in this case the inconsistencies are presented in the manipulator's x axis; therefore, this is the action dimension presented in the forthcoming plots. At each demonstration, the motion started from the region

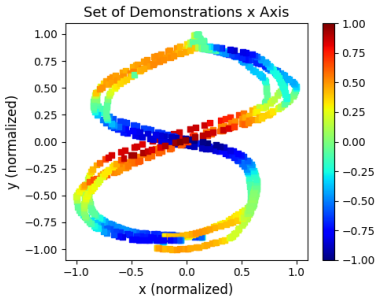
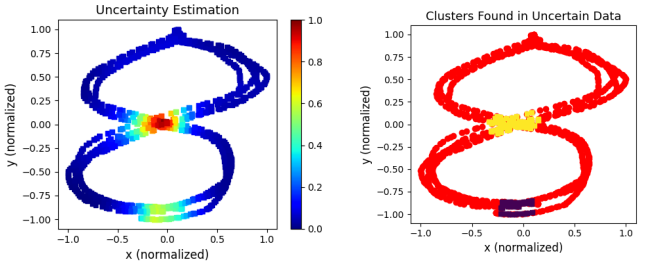


Fig. 8: Demonstrated trajectories in the XY plane. The color intensity of each point indicates the value of the demonstrated action in the x axis.



(a) Estimated aleatoric uncertainty of the data. The color of the data indicates the uncertainty computed at each point.

(b) Points in red represent the complete database and points in purple/yellow belong to the detected clusters.

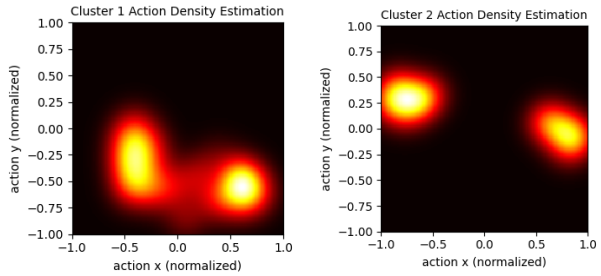
Fig. 9: Clustering of inconsistent events.

around the point (0.0, -0.9), which is also the a region with conflicting data. The other region with conflicting data is around the point (0.0, 0.0) and the goal is located roughly at (0.0, 0.9).

Once more, as expected, the aleatoric uncertainty estimated in the demonstrations was larger in the areas with conflicting data, as can be observed in Figure 9. It is also possible to observe that two clusters were obtained from DBSCAN (in purple and yellow, respectively), which is what we would expect to obtain as well.

In this case, it was required to estimate the probability density in the action space separately for each cluster. Figure 10 shows that, once again, a bi-modal distribution was predominant for each cluster, which is what we would expect to obtain from these demonstrations.

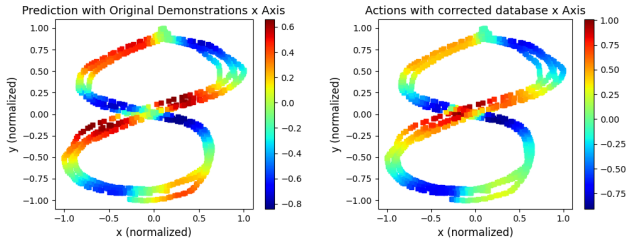
Finally, Figure 11 depicts how the aleatoric uncertainty was removed from the conflicting regions, and that one of the action choices was selected from each cluster



(a) Density estimation in the action space of cluster 1.

(b) Density estimation in the action space of cluster 2.

Fig. 10: Density estimation in the detected clusters.



(a) Neural Network regression using original demonstrations.

(b) Neural Network regression using modified demonstrations.

Fig. 11: Comparison of Neural Network regression before and after modifying the demonstrations.

to do so.

Twenty evaluations were done for each policy in this scenario as well. Once again, a trial was considered to be successful if the manipulator was able to reach the goal without colliding with the obstacles. In this case, the policy trained with the original demonstrations failed at every trial, while the policy trained with the modified demonstrations was successful at each one of them. It was observed that the former failed because it collided with one of the two obstacles at every trial.

V. Conclusions

We have introduced an algorithm to identify conflicting data in demonstration data sets and to automatically remove these conflicts through modifying this data, in scenarios involving equivalent action choices. This was achieved by applying uncertainty estimation with neural networks and processing the inference provided with clustering via the DBSCAN algorithm and selecting the actions to modify the examined data through kernel density estimation.

We have shown, through two distinct experiments, that this approach can lead to improved policies - the policies trained on data processed with our algorithm notably outperformed the policies trained on the unmodified data sets. In particular, our approach allowed for the utilization of data that prior to correction, did not lead to any successful evaluations. Furthermore, we have validated the assumptions taken within this approach and shown that the intermediate results are as expected.

Notwithstanding the successes of this algorithm, it is limited to scenarios involving equivalent action choices and its effectiveness and applicability to more challenging environments (e.g., higher-dimensional data or more complex ambiguities) remains to be explored. Finally, we note the possibility of combining our approach with active learning, such that the robot is able to query its teacher in regions of high uncertainty and thus resolve any conflicts with additional data.

References

[1] T. Osa, J. Pajarinen, G. Neumann, J. Bagnell, P. Abbeel, and J. Peters, "An Algorithmic Perspective on Imitation Learning," Foundations and Trends in Robotics, vol. 7, no. 1-2, pp. 1–179, 2018.

[2] A. G. Billard, S. Calinon, and R. Dillmann, "Learning from Humans," in Springer Handbook of Robotics, B. Siciliano and O. Khatib, Eds. Springer-Verlag GmbH, 2016, ch. 74. [Online]. Available: https://www.ebook.de/de/product/27988962/springer_handbook_of_robotics.html
[3] E. Billing and T. Hellström, "A Formalism for Learning from Demonstration," Paladyn, Journal of Behavioral Robotics, vol. 1, no. 1, pp. 1–13, jan 2010.
[4] Y. Gal, "Uncertainty in Deep Learning," Ph.D. dissertation, University of Cambridge, Sep. 2016.
[5] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight Uncertainty in Neural Networks," 2015.
[6] W. Clements, B. Robaglia, B. V. Delft, R. B. Slaoui, and S. Toth, "Estimating Risk and Uncertainty in Deep Reinforcement Learning," CoRR, vol. abs/1905.09638, 2019.
[7] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. Kochenderfer, "HG-DAgger: Interactive Imitation Learning with Human Experts," Oct. 2018.
[8] A. Kendall and Y. Gal, "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?" in Advances in Neural Information Processing Systems 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5574–5584.
[9] M. Segú, A. Loquercio, and D. Scaramuzza, "A General Framework for Uncertainty Estimation in Deep Learning," 2019.
[10] M. Tomczak, S. Swaroop, and R. Turner, "Neural network ensembles and variational inference revisited," in 1st Symposium on Advances in Approximate Bayesian Inference, 2018, pp. 1–11.
[11] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles," Dec. 2016.
[12] T. Pearce, M. Zaki, A. Brintrup, and A. Neely, "Uncertainty in Neural Networks: Bayesian Ensembling," Oct. 2018.
[13] D. Nix and A. Weigend, "Estimating the mean and variance of the target probability distribution," in Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN '94). IEEE, 1994.
[14] M. Ayhan and P. Berens, "Test-time data augmentation for estimation of heteroscedastic aleatoric uncertainty in deep neural networks," 2018.
[15] P. Becker, O. Arenz, and G. Neumann, "Expected information maximization: Using the i-projection for mixture density estimation," in International Conference on Learning Representations, 2020.
[16] L. Ke, M. Barnes, W. Sun, G. Lee, S. Choudhury, and S. S. Srinivasa, "Imitation learning as f-divergence minimization," CoRR, vol. abs/1905.12888, 2019.
[17] S. Nowozin, B. Cseke, and R. Tomioka, "f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization," in Advances in Neural Information Processing Systems, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016.
[18] S. Ledesma, M.-A. Ibarra-Manzano, E. Cabal-Yepez, D.-L. Almanza-Ojeda, and J.-G. Avina-Cervantes, "Analysis of Data Sets With Learning Conflicts for Machine Learning," IEEE Access, vol. 6, pp. 45 062–45 070, 2018.
[19] S. Chernova and M. Veloso, "Learning Equivalent Action Choices from Demonstration," in 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, Sep. 2008.
[20] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-based Algorithm for Discovering Clusters a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, ser. KDD'96. AAAI Press, 1996, pp. 226–231. [Online]. Available: http://dl.acm.org/citation.cfm?id=3001460.3001507
[21] C. Bishop, Pattern Recognition and Machine Learning. New York: Springer, 2006, ch. 2.5, pp. 120–127.