Reasoning and Learning in Real-World Systems for Long-Term Autonomy (LTA)
Papers from the AAAI 2018 Fall Symposium

Risk-Aware Planning by Extracting Uncertainty from Deep Learning-Based Perception

Maymoonah Toubeh and Pratap Tokekar
Virginia Polytechnic Institute and State University
Bradley Department of Electrical and Computer Engineering
Blacksburg, Virginia 24061

Abstract

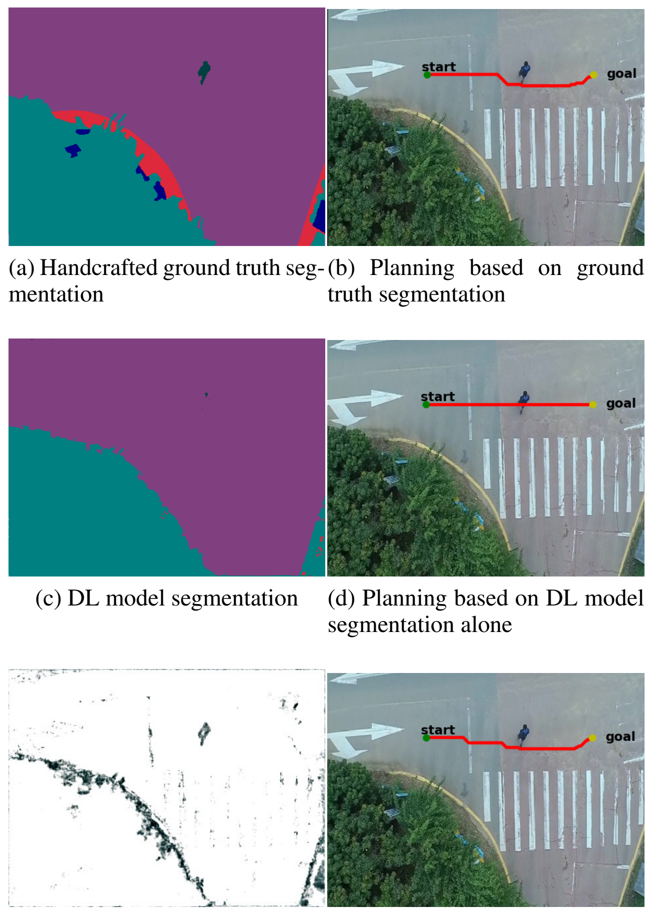
The integration of deep learning models and classical techniques in robotics is constantly creating solutions to problems once thought out of reach. The issues arising in most models that work involve the gap between experimentation and reality, with a need for a quantification of risk in real-world situations. In order to translate advances in robot planning techniques that use deep learning to safety-critical applications, strategies must be developed and applied to assess the risk involved with different models. This work proposes the use of Bayesian approximations of uncertainty from deep learning in a robot planner, showing that this produces more cautious actions in safety-critical scenarios. An example setup involving a deep learning semantic image segmentation, followed by a path planner based on the resulting cost map, is used to provide an empirical analysis of the proposed method.

Introduction

Recent advances in deep learning (DL) training algorithms, paired with significant improvements in hardware, have shown potential in many fields, including robotics. From enabling robot systems to navigate using high-dimensional image inputs, to allowing tractable trial-and-error robot learning both in simulation and reality; deep learning is everywhere. However, as promising as applications of DL to robot planning seem, the potential of the positive impact they may have on real-world scenarios is inevitably proportionate to their interpretability and applicability to imperfect environments.

For an example setup, this work utilizes a DL image segmentation model to generate a cost map used in an A* path planner. Figure 1 shows qualitative results given by the DL model and the subsequent planner. In this image taken from the Aeroscapes dataset (Nigam, Huang, and Ramanan 2018), the pedestrian near the top center is not sufficiently segmented by the DL model prediction shown in Figure 1a. Incorporating uncertainty associated with this prediction before passing it to the planner produces a more reasonable and risk-aware resulting path, as seen in Figure 1f. Higher levels of uncertainty are visualized by darker spots in Figure 1e.

DL is known for its data-driven rather than algorithmic learned representations (LeCun, Bengio, and Hinton 2015). A DL hierarchical structure can learn directly from data with



(a) Handcrafted ground truth segmentation
(b) Planning based on ground truth segmentation
(c) DL model segmentation
(d) Planning based on DL model segmentation alone
(e) Uncertainty of DL model segmentation
(f) Planning based on DL model segmentation with uncertainty

Figure 1: Qualitative results showing the path planned from start to goal given (a) handcrafted ground truth image segments, (b) deep learning model segmentation alone, (c) deep learning model segmentation with uncertainty.

little to no handcrafted features or learning variables (Goodfellow, Bengio, and Courville 2016). However, this often comes at the expense of the interpretability of the learning outcomes. Deep neural networks can even misrepresent data outside the training distribution, giving predictions that are incorrect without providing a clear measure of certainty associated with the result (Gal 2017). The outputs of a deep neural network are generally point estimates of the parameters and predictions present, so they do not provide a meaningful measure of correlation to the overall data distribution the network was trained on (Gal 2017). For this reason, deep learning models are considered deterministic functions, often called "black-boxes", unlike probabilistic models which inherently depict uncertainty information.

As a step towards risk-aware robotic systems that utilize the powers of DL, this work combines methods of approximating uncertainty in DL with robot planners that are partially reliant on an otherwise black-box approach. We utilize modern methods of uncertainty extraction from deep learning models, specifically those that do not interfere with the overall structure or training process (Gal and Ghahramani 2016). The extraction of uncertainty information, as opposed to the reliance on point estimates, is crucial in safety-critical applications, such as autonomous navigation in an urban setting. If a robot planner encounters out-of-distribution data at test time, it is preferable that the system provides an uncertainty metric to allow for more meaningful interpretation of a forced point estimate. With the acquired Bayesian uncertainty estimates, the system produces an explicable metric and can therefore be altered to accommodate for risks in the environment.

In the robotics community, an emphasis has been placed on methods that work in a controlled experimental setup, but more recently risk-aware methods aim to ensure that these methods are also safe in the real-world. As a relatively new application to robotics, techniques have been adapted from the fields of statistics and machine learning. Common statistical methods of accommodating risk include altering the optimization criterion so that it becomes risk-sensitive (Garca and Fernández 2015). Although modern DL models are usually considered black-boxes due to their mathematical nature, recent work has initiated theoretically grounded understandings of them. Such works investigate the integration of deep learning techniques with information theoretical and statistical approaches for the purpose of calculating model uncertainties (Gal 2017). It is these practical methods of quantifying risk associated with DL models that are utilized in the proposed planning systems of this paper.

Related Work

Several prior works exist which have extracted uncertainty information from a deep learning model, mainly Bayesian neural networks, ensemble methods, and methods that utilize stochastic regularization techniques (Gal 2017; Osband et al. 2016). These works have also produced meaningful contributions to real-world applications, such as gas turbine control, camera relocalization, and robotic collision avoidance (Depeweg et al. 2016; Kahn et al. 2017; Kendall and Cipolla 2016). Our work seeks to extend the most suitable of these approaches to robot planning.

Bayesian Neural Networks

Some of the earliest attempts to bind the reasoning of probabilistic models, such as Gaussian processes, with deep learning (DL) is seen in Bayesian neural networks (BNN) (Gal 2017; Kononenko 1989). Unlike the DL models used in modern practice which do not depict uncertainty, a BNN produces an output that is a probability distribution over its predictions. Probability distributions are placed over the weights of a BNN, making it an approximation of a Gaussian process as the number of weights tends to infinity. Uncertainty can be extracted as a statistical measure, such as variance or entropy, over this output distribution in order to capture how confident the model is with its prediction. However, BNN require a larger number of parameters to be trained, with less practical training methods available for them. A recent example of a Bayesian neural network applied to a stochastic dynamic system utilizes a smaller model and trains by minimizing alpha-divergences (Depeweg et al. 2016). The system dynamics are learned using the BNN and are fed into a model-based reinforcement learner for control, with application to a gas turbine.

Ensemble Methods

Since the practicality of Bayesian neural networks is questionable, approximations of these structures have arisen, including ensemble methods. One such recent method is referred to as the bootstrapped neural network, where several deep learning (DL) models are trained on subsets of the larger dataset sampled with replacement. The underlying concept behind this method is that the different models will agree in high density areas and disagree in low density areas of the complete dataset. The outputs of the separate models combine to form a probability density function from which uncertainty of a particular prediction can be measured. This method is theoretically sound; however, it is not ideal for applications that are faced with time and resource constraints, such as robotics.

A recent work proposes using bootstrapped deep Q-learning networks quantifying uncertainty to direct the learning process towards more efficient exploration, which is a key issue in reinforcement learning (Osband et al. 2016). The technique involves sampling the past experiences in Q-learning, rather than taking the whole sequence, in order to form an estimate of the Q-value at a given state. Sampling is also meant to scale better to large state-spaces. A similar approach is used to improve a form of Q-learning, Deep Deterministic Policy Gradient (DDPG) (Kalweit and Boedecker 2017). The work demonstrates that using bootstrapped uncertainty to direct data collection produces faster learning that is also less expensive, tedious, or likely to lead to physical damage for a real robot.

Stochastic Regularization Methods

Most recently, the use of stochastic regularization methods common in deep learning (DL) has been shown to also

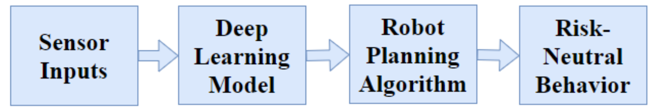
Class	Sky	Building	Pole	Road	Pavement	Tree	Sign Symbol	Fence	Car	Pedestrian	Bicyclist	Unlabeled
Cost	15	11	10	1	2	7	9	8	12	14	13	16
Color												

Table 1: The fixed costs assigned to each segmentation to be used in A* search. These costs are hand-designed to generate qualitative examples that demonstrate the utility of the proposed approach. In the real world, classes that are not navigable (e.g., fence, car, bicylist) will be assigned infinite cost.

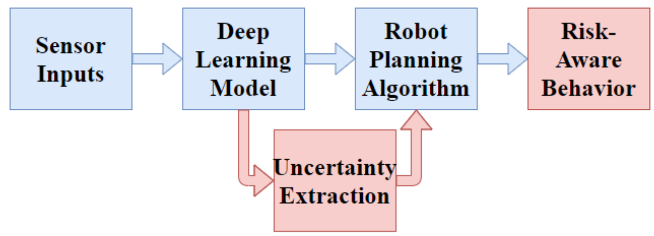
approximate Bayesian neural network models without any changes to the ready structures being used or their training process (Gal 2017). Regularization methods are used as a means to avoid overfitting of a DL model to its training set, so that it generalizes to data that is similar enough but not exactly the same. This ensures the learning model has not simply memorized the training data, but has actually learned something meaningful about the data that will translate to a slightly different setting. At a high level, stochastic regularization techniques work by introducing randomness in the training process to increase the robustness of the model to noise. Dropout is one such popular method that is inspired by the probabilistic interpretations of deep learning models that consider activation nonlinearity a cumulative distribution function (Bishop 2006). In its traditional use, dropout is activated during training, in which case the weights of a deep learning model are randomly multiplied by zero or one in a certain predefined proportion. In this approach, at test time, weight averaging by the percentage of dropout applied during training is performed on the trained model, which then leads to point estimate results.

In order to form a distribution over the outputs of a model trained using a stochastic regularization technique such as dropout, the regularization is activated at test time, producing stochastic estimates with multiple passes of the same input through the model (Gal and Ghahramani 2016). The multiple stochastic passes are then averaged to form a mean estimate, as opposed to a point estimate, and uncertainty can be extracted given the statistical or information theoretical metrics over the distribution.

One recent work applies Bayesian approximation using dropout to a pre-existing model-based reinforcement learning algorithm, called PILCO, in order to better quantify uncertainty over longer periods of learning (Gal, McAllister, and Rasmussen 2016). The authors replace the original Gaussian processes with a deep neural network. In another work, the Bayesian approximation of uncertainty with dropout is used to assist in camera relocalization for landmark detection in a SLAM problem (Kendall and Cipolla 2016).The uncertainty estimate is used to approximate the localization error with no additional hand-crafted parameterizations. In a similar work, the same Bayesian approximation approach to uncertainty is applied to semantic segmentation for improved learning and test time estimation (Kendall, Badrinarayanan, and Cipoll 2015). In yet another work, the authors combine bootstrapped neural networks with stochastic regularization methods to avoid catastrophic or harsh collisions during robot training for collision avoid-



(a) Risk-Neutral Perception



(b) Risk-Aware Perception

Figure 2: The overall flow of information in a robot planner that relies in part on a DL model in (a) the risk-neutral and (b) the risk-aware case.

ance (Kahn et al. 2017). They show that their method effectively minimizes dangerous collisions during training, while also showing comparable performance to baselines without explicit account for uncertainty. A more recent approach proposes the use of a Mixture Density Model (MDM) as a replacement for the stochastic passes, where a single pass saves time in comparison to multiple in a robotics setting (Choi et al. 2017). However, this comes at the expense of training a separate MDM network for the task of uncertainty extraction.

Problem Formulation

The problem setup is inspired by a previous work (Christie et al. 2017) that uses the overhead imagery provided by an unmanned aerial vehicle (UAV) as input to an image segmentation algorithm, which is then used to assist the navigation of an unmanned ground vehicle (UGV). The UAV acts as a “scout” by flying ahead of the UGV. The overhead orthorectified imagery is then classified (into categories such as “road”, “pavement”, “car”, etc.). Each category is assigned a cost (given in Table 1) which is used to determine a path for the ground robot to follow.

Figure 2 shows a high-level schematic of the proposed risk-aware approach by contrasting it with the traditional, risk-neural approach. Unlike the previous work, here, in addition to performing the semantic segmentation of the image, the uncertainty in the segmentation is also extracted. The measure of uncertainty is then used to manipulate the navigation away from low confidence regions. The navigation portion of the robot planner in this case is not a DL model, but a classical method, A* search. A cost function is mapped onto each semantic class (see Table 1). Therefore, the uncertainty in the segmentation corresponds to uncertainties in the cost, for which A* search is sufficient. If the uncertainties in the transition function are to be considered instead, a Markov Decision Process (MDP) would be more useful. Considering uncertainty is contrast to trusting the outputs of the DL portion of the system invariably, which could lead to catastrophic outcomes if a point estimate outlier is produced in the case the input is considered out-of-distribution to the training data. In the proposed approach, an uncertainty metric can be used to calibrate the robot plan based on the level of confidence in the DL model predictions.

The results of the segmentation given by any DL model cannot guarantee complete accuracy in all settings. Variations in lighting, angle, or objects present in an image can contribute to inaccurate predictions. There will always be a prediction when a DL model is involved, as the model will force an estimate even when it does not make sense to. A good measure to account for this risk associated with DL outputs being used in the robot planner is to evaluate the certainty associated with the DL result. One practical method is using dropout, which is already being used as a regularizer during training.

In the risk-neutral case, the pixel classification is taken as is from the DL model and assigned a cost accordingly. For the proposed risk-aware method, the cost is evaluated by adding the uncertainty value, multiplied by some factor, to the risk-neutral cost assignment. Specifically, the cost of pixel p is given by,

$$C(p) = L(p) + \lambda \hat{V}(p), \tag{1}$$

where $L(p)$ is the cost associated with the semantic class that is predicted for p (given in Table 1) and $\hat{V}(p)$ is the uncertainty value extracted by dropout. In practice, $\hat{V}(p)$ does not need to be variance; it can be another statistical measure of uncertainty taken over the distribution provided by the stochastic passes. λ is a weighting parameter. The risk-neutral case corresponds to $\lambda = 0$ and the risk-aware case corresponds to $\lambda > 0$.

Algorithm 1 shows a breakdown of uncertainty extraction given an input image. First, the stochastic outputs are generated for a number of stochastic passes, giving a softmax value for each pixel each time. For each pass, Bernoulli dropout is activated on the trained network, effectively multiplying random neuron weights by zero or one in a set proportion. The softmax outputs O are averaged over all stochastic passes. The maximum value of this average softmax is taken as the output class prediction Ind. Variance is computed over the stochastic passes for each output class, then the average of V over all classes produces a single value for each pixel. The value $\hat{V}$ is considered the uncertainty in that pixel's prediction.

Algorithm 1 Uncertainty Extraction

Input: image I
Output: class of average prediction for each pixel $Ind(p)$, average variance for each pixel $\hat{V}(p)$

$p \leftarrow$ pixels in image I
for t = 1 to number of stochastic passes **do**
  $O(p, c, t) \leftarrow$ softmax output of stochastic pass t
end for
$\hat{O}(p, c) \leftarrow$ average $O(p, c, t)$ over stochastic passes
$Ind(p) \leftarrow$ argmax of softmax in $\hat{O}(p, c)$
for c = 1 to number of classes **do**
  $V(p, c) \leftarrow$ variance in $O(p, c, t)$ over stochastic passes for class c
end for
$\hat{V}(p) \leftarrow$ average $V(p, c)$ over classes

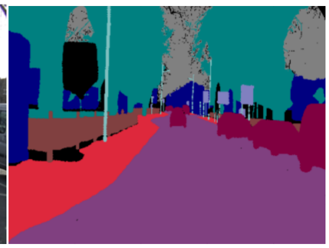
The uncertainty value is computed as the average variance across all segment classes for a particular pixel. The higher the average variance, the less confident the DL model is with its prediction. Therefore, it is intuitive to incorporate this information along with the original prediction when planning a path for navigation, especially in a safety-critical environment such as a road.

Experimental Setup

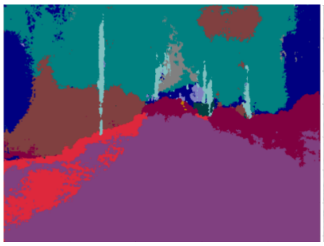
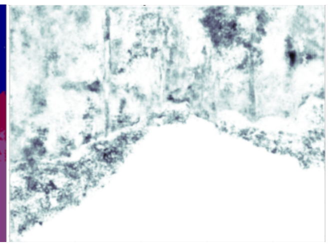
To demonstrate the utility of an uncertainty metric associated with a DL model being used as part of a robot planning system, experiments involving this setup are portrayed to compare the risk-neutral and risk-averse cases qualitatively.

We use the Bayesian SegNet to perform semantic segmentation of every pixel in the input image (Kendall, Badrinarayanan, and Cipoll 2015). The Bayesian SegNet is first trained for the segmentation task using the predefined model architecture, along with the pretrained weights of the VGG16 image classification network (Kendall, Badrinarayanan, and Cipoll 2015). Since the model is already well adapted for image classification, less further training is needed for per-pixel segmentation, in comparison to starting with random model weighting. The CamVid dataset (Brostow, Fauqueur, and Cipolla 2009) of 367 training and 233 testing images of road scenes is used. The model converges with a test accuracy of 94.56 %. A batch size of 2 is used to fit an 8GB Nvidia GeForce GTX 1080. Some results of this network are shown in Figure 3 for the CamVid test set, as well as for an entirely different road dataset called KITTI (Geiger et al. 2013) in Figure 4. It is worth noting that the KITTI dataset is an example for which the DL model is not explicitly trained, but its images are similar enough to the training set to expect reasonable segmentation performance. Errors are common, and expected, in a varying or real-world setting, but the uncertainty metric should provide a means of detecting such errors.

After the DL model is trained to produce reasonable results on images similar to the training set, the next step involves using the pixel segmentations as input to the A*

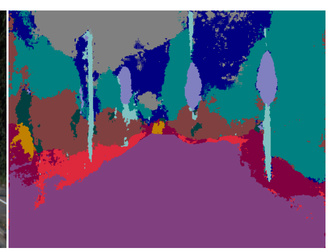



(a) Input road image (b) Handcrafted ground truth segmentation

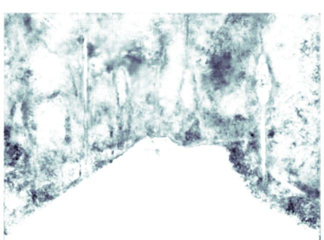



(c) DL model segmentation (d) DL model uncertainty

Figure 3: An example semantic segmentation produced by the Bayesian SegNet trained model on (a) an input image from the CamVid test set with (b) its handcrafted ground truth segmentation, alongside (c) the resulting output segmentation and (d) the model uncertainty associated with the output, with darker regions signifying higher uncertainty.

(a) Input road image (b) DL model segmentation



(c) DL model uncertainty

Figure 4: An example semantic segmentation produced by the Bayesian SegNet trained model on (a) an input image from the KITTI dataset with (b) the resulting output segmentation and (c) the model uncertainty associated with the output, with darker regions signifying higher uncertainty.

Measure	Ground Truth	Risk-Neutral	Risk-Aware
Expected Cost	251	234	253
Actual Cost	251	413	281
Surprise Factor	0	179	28

Table 2: The surprise factor calculated as the difference between the expected cost and actual cost for the example in Figure 1, given ground truth, risk-neutral, and risk-aware segmentation.

search planner. In order to use semantic segmentation in robot planning, a cost function must be defined over the segmentation result to be used in the planning. The most common method of translating segment identity into a real number cost involves some quantification of the segmentation class traversability combined with the robot's speed. For example, in the previous work inspiring our problem setup, segment traversability is proportionate to the power consumption of the UGV based on prior experience (Christie et al. 2017). However, for simplicity, here, the costs of the segment classes are set to a fixed value, as shown in Table 1. Once a cost is assigned, a start position and a goal position are defined in the input image, and the planning algorithm produces a path based on the cost assigned to the predicted pixel identification.

Results and Discussion

In order to perform a qualitative analysis of the risk-neutral and risk-aware methods, the *surprise factor* is calculated to compare the expected path cost with the actual path cost by subtracting the two. The expected path cost is found by summing up the cost associated with every pixel along the path. We use the predicted class of every pixel to determine the expected cost and the ground truth class of every pixel to determine the actual cost. If a path passes through pixels that the DL model classifies with low uncertainty, then we expect the predicted classes to be largely the same as the actual cost, thereby given a low surprise factor. On the other hand, if the predicted classes are wrong, then the surprise will be high.

Table 2 shows this factor in three planning scenarios of Figure 1: using ground truth segmentation, using DL model segmentation alone, and using DL model segmentation while taking into account its prediction uncertainty. When using DL segmentation alone, not accounting for model confidence increases the chance for a higher disparity between the expectation and reality, as seen in our example. On the other hand, although the risk-aware approach results in longer paths, the expectation better matches the reality and leads to lower surprise.

Table 3 shows a similar trend for the average surprise factor calculated over 100 randomly chosen starting and goal positions. In this case, the risk-neutral strategy tends to underestimate the actual path cost resulting in a larger surprise factor. On the other hand, the risk-aware strategy conservatively chooses longer paths but gives a significantly lower surprise factor.

Measure	Ground Truth	Risk-Neutral	Risk-Aware
Expected Cost	1221.60	1160.10	1427.40
Actual Cost	1221.60	1480.10	1405.35
Surprise Factor	0.00	320.00	22.05

Table 3: The average surprise factor calculated as the difference between the expected cost and actual cost for 100 randomly chosen starting and goal positions, given ground truth, risk-neutral, and risk-aware segmentation.

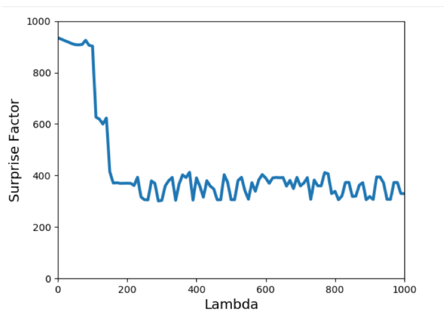


Figure 5: Effect of varying λ on the surprise factor.

Figure 5 shows the effect of varying the λ parameter on the surprise factor. When λ is small, the surprise factor is large. This is consistent with previous findings, since $\lambda = 0$ corresponds to the risk-neutral case. As λ increases, the surprise factor decreases finally converging to a fixed value. This is because, once λ is sufficiently large, increasing λ further does not change the path produced as output significantly (except for a few pixels). In fact, for very large λ, the path found will be the minimum uncertainty path since the second term in Equation 1 dominates the first term. Therefore, the surprise factor remains largely the same.

Conclusions and Future Work

This work proposes a risk-aware approach to robot planning that already involves deep learning. Risk is quantified by the model prediction uncertainty in the planning process. When deep learning is used for perception as a portion of the planning loop, an understanding of confidence in DL estimates is useful. Uncertainty is extracted directly from the DL models utilizing dropout as a practical method, especially in resource constrained settings such as robotics. Promising results show that including uncertainty in a planner provides better predictability of actions, and even the avoidance of catastrophic actions in a safety-critical setting. Further experiments and empirical analysis are to be conducted using statistically significant data to explore the potential of this method for real-world systems in the long run. Some scenarios include those presented in this paper, such as the UAV assisted navigation of a UGV using DL image segmentation, on hardware and photorealistic simulators. The main anticipated extensions of this work involve the use of a larger dataset analysis, along with investigating different uncertainty metrics extracted from the same experiment setup presented in this paper.

Acknowledgements

The material is based upon work supported by the National Science Foundation under Grant number 1566247.

References

Bishop, C. M. 2006. *Pattern Recognition and Machine Learning*. New York: Springer.

Brostow, G. J.; Fauqueur, J.; and Cipolla, R. 2009. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters* 30(2):88–97.

Choi, S.; Lee, K.; Lim, S.; and Oh, S. 2017. Uncertainty-aware learning from demonstration using mixture density networks with sampling-free variance modeling. *arXiv preprint arXiv:1709.02249*.

Christie, G.; Shoemaker, A.; Kochersberger, K.; Tokekar, P.; McLean, L.; and Leonessa, A. 2017. Radiation search operations using scene understanding with autonomous uav and ugv. *Journal of Field Robotics* 34(8):1450–1468.

Depeweg, S.; Hernández-Lobato, J. M.; Doshi-Velez, F.; and Udluft, S. 2016. Learning and policy search in stochastic dynamical systems with bayesian neural networks. *arXiv preprint arXiv:1605.07127*.

Gal, Y., and Ghahramani, Z. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, 1050–1059.

Gal, Y.; McAllister, R.; and Rasmussen, C. E. 2016. Improving pilco with bayesian neural network dynamics models. In *Data-Efficient Machine Learning workshop, ICML*.

Gal, Y. 2017. *Uncertainty in Deep Learning*. Ph.D. Dissertation, University of Cambridge.

Garca, J., and Fernández, F. 2015. A comprehensive survey on safe reinforcement learning. In *Journal of Machine Learning Research*, volume 16, 1437–1480.

Geiger, A.; Lenz, P.; Stiller, C.; and Urtasun, R. 2013. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research*.

Goodfellow, I.; Bengio, Y.; and Courville, A. 2016. *Deep Learning*, volume 1. Cambridge, MA: MIT Press.

Kahn, G.; Villaflor, A.; Pong, V.; Abbeel, P.; and Levine, S. 2017. Uncertainty-aware reinforcement learning for collision avoidance. *arXiv preprint arXiv:1702.01182*.

Kalweit, G., and Boedecker, J. 2017. Uncertainty-driven imagination for continuous deep reinforcement learning. In *Conference on Robot Learning*, 195–206.

Kendall, A., and Cipolla, R. 2016. Modelling uncertainty in deep learning for camera relocalization. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, 4762–4769. IEEE.

Kendall, A.; Badrinarayanan, V.; and Cipoll, R. 2015. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*.

Kononenko, I. 1989. Bayesian neural networks. *Biological Cybernetics* 61(5):361–370.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *Nature* 521(7553):436.

Nigam, I.; Huang, C.; and Ramanan, D. 2018. Ensemble knowledge transfer for semantic segmentation. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1499–1508. IEEE.

Osband, I.; Blundell, C.; Pritzel, A.; and Van Roy, B. 2016. Deep exploration via bootstrapped dqn. *Advances in Neural Information Processing Systems* 4026–4034.