Deep Networks with Confidence Bounds for Robotic Information Gathering

Jeffrey A. Caley, Nicholas R.J. Lawrance and Geoffrey A. Hollinger

***Abstract*—We propose a convolutional LSTM network with bootstrapped confidence bounds as a method for modeling spatio-temporal data. By providing estimates with confidence bounds that are accurate far into the future, multi-step planners can be utilized to improve performance on information gathering missions. This technique is compared to existing environment modeling techniques. We demonstrate that our proposed approach constructs long horizon estimates with greater accuracy. We also achieve more accurate and more conservative confidence bounds. Validation through simulation shows our technique increases path planning performance in environmental information gathering missions.**

I. Introduction

The capability of mobile robots to collect data over large spatial and temporal scales has lead to their increasing use for information gathering tasks. Applications include marine monitoring [6], atmospheric modeling [21], and search and rescue [2]. In many cases, the value of data collected during a deployment is dependent on the conditions in which the data was collected. Planning for maximizing informative samples is an ongoing problem for mobile robots.

Our work focuses on the domain of monitoring marine ecosystems. Marine ecosystems change and evolve over a wide range of spatial and temporal scales, making collection of scientific data especially challenging due to the expense of long-duration ocean surveys that traditionally required deployments from manned surface vessels. Fortunately, the introduction of autonomous ocean gilders provides a lower-cost alternative for collecting data over long time periods [23].

Previously, robots were often deployed using pre-generated search plans such as boustrophedon (lawnmower) paths that provide reliable spatial coverage but do not optimize other information measures [4]. Estimating the information value of future samples is challenging at the planning stage as it requires estimating the future state of the environment. One approach is to learn from prior data. In some cases, satellite data provides historical and current estimates of environmental conditions. While there are many possible approaches to learning models for prediction, we are interested in model-free methods that also provide probabilistic confidence estimates. Model-free (data-driven) approaches provide improved flexibility for different input data types and are applicable over a wider range of domains. Confidence bounds provide additional information that can improve planning performance and allow for strategies that actively maximize information for targeted exploration [20].

The method presented in this paper is based on two primary components: a deep neural network to generate environment predictions, and a Monte Carlo Tree Search (MCTS) planner to generate plans. We propose a recurrent convolutional neural network to capture temporal and spatial correlation in ocean data. By training multiple parallel networks we also generate confidence bounds from the network predictions. We also propose an associated MCTS planner that uses the neural network predictions and associated confidence bounds to generate plans that maximize data collected in targeted conditions. Using historical ocean data, we demonstrate significant gains in prediction quality which translate to modest gains in planning performance. The key novelty of this paper is the development of bootstrap confidence bounds for convolutional LSTM networks. This results in powerful long horizon estimates with confidence bounds that outperform existing Gaussian process methods.

II. Related Work

A. Artificial Neural Networks

1) Convolutional Neural Networks: Convolutional neural networks (CNNs) are a type of artificial neural networks where nodes are connected in specific configurations to take advantage of spatially related data [14]. Work by Levine et al. shows how CNNs can be used in robotic applications [16]. In this work, a CNN is utilized to recognize an end effector and target location and directly map those predictions to motor commands. This end-to-end system was able to perform simple tasks, such as putting a cap on a bottle or hanging a coat hanger on a rod using only a single neural network. This was one of the first examples of a CNN used for solving robotic problems. Further work by Levine et al. also demonstrated a CNN used for grasping [17]. In this work they used data collected from over 800,000 grasp attempts to train a CNN to estimate the success rate given an object position and grasping strategy.

2) Recurrent Neural Networks: Recurrent neural networks (RNN) are a type of artificial neural network that take advantage of sequential data. Long Short Term Memory networks (LSTM) are a special kind of RNN that use a form of memory to recognize long term trends in data. Convolutional LSTMs are introduced in [25] to take advantage of the spatial correlations in spatio-temporal sequences by using an LSTM network configuration with convolutional structures in the

This work has been funded in part by the Office of Naval Research grant N00014-14-1-0509.

J. Caley, N. Lawrance and G. Hollinger are with the Robotics Program, School of Mechanical, Industrial and Manufacturing Engineering, Oregon State University {caleyj, nicholas.lawrance, geoff.hollinger}@oregonstate.edu.

LSTM network. They show this configuration outperforms the traditional fully connected LSTM network on spatio-temporal data. The spatio-temporal model learning used in this work is directly related to the work we present here.

B. Informative Planning Problems

Informative path planning has a history of using traditional path planning problems and adaptive sampling techniques to guide their work. For problems where the search space is small, A* can be used to solve for the optimal path in discretized spaces [8]. Heuristic search [13], random graph search [19], and sampling based approaches [12] [9] have all been used when the search space grows large and a comprehensive search becomes impossible. A branch and bound technique is used to solve the optimal path in [1], but requires a complete map of the world and can become intractable at very large map sizes. Monte Carlo Tree Search (MCTS) [3] is a method for making decisions in a given domain by taking smart samples in the decision space and building a search tree according to the results. MCTS has been used to play many games successfully, such as Ms Pac-Man [22], Go [15] and the card game Magic The Gathering [24]. Monte Carlo Tree search lends itself to large spatio-temporal planning problems and plays an important roll in this work.

III. Problem Formulation

In this work, we investigate an agent as it performs an information gathering mission. The world is considered as a three-dimensional space $\chi \in \mathbb{R}^3$ (two spatial dimensions and time) discretized into a regular grid in space and time. The goal of the agent is to collect observations at locations in the environment $\mathbf{x} \in \chi$ that fall within a specified range of a target variable $z(\mathbf{x}) : \mathbb{R}^3 \to \mathbb{R}$. We assume that the agent has access to historical and recent data of the target variable, but does not have any future knowledge. To navigate the world, at each time step the agent's planner selects a movement action which moves the agent one unit (of the grid resolution) in one of the four cardinal compass directions, and one unit forward in time. A plan P is defined as a finite sequence of visited states, $P = \{\mathbf{x}_0, \ldots, \mathbf{x}_n\}$. The agent is rewarded using a binary score function $R(\mathbf{x})$, that takes the value 1 for each time step it records data in a (previously unobserved) location where the target variable falls within the specified range, $z(\mathbf{x}) \in [z_{min}, z_{max}]$. The agent otherwise receives no score,

$$R(\mathbf{x}_k) = \begin{cases} 1 & z(\mathbf{x}_k) \in [z_{min}, z_{max}], \mathbf{x}_k \notin P_{0:k-1}, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

In the current work, the target variable z is the ocean temperature, but the method is not limited to this application. The objective for the planner is to generate a plan P^* from the current time $t = 0$ to a limited horizon t_n that maximizes the objective R (Eq.(2)) along the length of the path,

$$P^* = \arg\max_P \sum_{t=0}^{t_n} R(\mathbf{x}_t). \quad (2)$$

IV. Method

A. Deep Network with Confidence Bounds

The proposed method creates a prediction model for spatio-temporal data with confidence bounds through the use of bootstrapped convolutional LSTM (CNN-LSTM) networks. This novel architecture includes the training of multiple CNN-LSTM networks to construct a model that can predict multiple values accuracy in advance and provide conservative confidence bounds on those estimates.

1) ConvLSTM: Long Short-Term Memory (LSTM) is a type of recurrent neural network with built in memory cells to store information. This architecture addresses the problem of long-term dependency problems, where older information is forgotten as new information arrives. A memory cell c_t acts as an accumulator of state information. As new information comes in, an input gate i_t will decide whether to accumulate that information into the memory cell. Old cell information c_{t-1} can be removed if the gate f_t is active, often called the forget gate. The key equations for the formulation of a fully connected LSTM are shown in (3) through (7), where $\circ$ denotes the Hadamard product. This follows the same formulation as seen in [7].

$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci} \circ c_{t-1} + b_i\right) \quad (3)$$
$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf} \circ c_{t-1} + b_f\right) \quad (4)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (5)$$
$$o_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co} \circ c_t + b_o\right) \quad (6)$$
$$\mathcal{H}_t = o_t \circ \tanh(c_t) \quad (7)$$

The convolutional LSTM augments the traditional LSTM network to take advantage of spatial data. This is accomplished by making the inputs $\mathcal{X}_t$, cell outputs $\mathcal{C}_t$, hidden states $\mathcal{H}_t$ and gates 3D tensors with spatial dimensions. Effectively combining the spatial advantages of a convolutional neural network with the temporal advantages of an LSTM network. The key equations for ConvLSTM are shown in (8) through (12) (the * denotes a convolution operator).

$$i_t = \sigma\left(W_{xi} * \chi_t + W_{hi} * \mathcal{H}_{t-1} + W_{ci} \circ \mathcal{C}_{t-1} + b_i\right) \quad (8)$$
$$f_t = \sigma\left(W_{xf} * \chi_t + W_{hf} * \mathcal{H}_{t-1} + W_{cf} \circ \mathcal{C}_{t-1} + b_f\right) \quad (9)$$
$$\mathcal{C}_t = f_t \circ \mathcal{C}_{t-1} + i_t \circ \tanh(W_{xc} * \chi_t + W_{hc} * \mathcal{H}_{t-1} + b_c) \quad (10)$$
$$o_t = \sigma\left(W_{xo} * \chi_t + W_{ho} * \mathcal{H}_{t-1} + W_{co} \circ \mathcal{C}_t + b_o\right) \quad (11)$$
$$\mathcal{H}_t = o_t \circ \tanh(\mathcal{C}_t) \quad (12)$$

2) Deep Network Estimate: The CNN-LSTM network used for prediction is trained using sequential frames of 50×50 pixel images with values normalized between 0 and 1 representing ocean temperatures. The network outputs a 50×50 image with corresponding estimates of temperature for the next frame in the video. For the CNN-LSTM architecture used in our work, see Fig. 1. The error function used for training is mean square error (MSE). Three ConvLSTM layers

were selected through a process of ad-hoc testing of multiple layer combinations and filter values. The single filter 3D convolutional layer at the end combines the data back into the correct output dimensions.

3) Bootstrap Method for Confidence Bounds: When constructing confidence bounds for neural networks (NN), the variance of a point in the model can be represented as

$$\sigma_i^2 = \sigma_{\hat{y}_i}^2 + \sigma_{\hat{\varepsilon}_i}^2 \tag{13}$$

where $\sigma_{\hat{y}_i}^2$ comes from the model misspecification and parameter estimation error associated with the modeling neural network [11]. $\sigma_{\hat{\varepsilon}_i}^2$ is considered the measure of noise variance. By constructing an estimate of both of these values, the total variance can be estimated and confidence bounds can be constructed.

One technique for predicting these values is the bootstrap method [11]. The bootstrap method assumes that an ensemble of NN models will generate a less biased estimate due to the NN models residing in different subsets of the parameter space. This allows us to construct B NN models and use those to estimate the value of $\sigma_{\hat{y}_i}^2$

$$\sigma_{\hat{y}_i}^2 = \frac{1}{B-1}\sum_{b=1}^{B}\left(\hat{y_i^b} - \hat{y}_i\right)^2 \tag{14}$$

where $\hat{y_i^b}$ is the prediction of the ith sample generated by the bth boostrap model and $\hat{y}_i$ is the true value of the ith sample. This variance can be attributed to the random initialization of each NN in the ensemble and the sampled data sets used for training the NNs.

To get an estimate of $\sigma_{\hat{\varepsilon}_i}^2$ we need to estimate the variance of errors. From (13) $\sigma_{\hat{\epsilon}}^2$ can be calculated:

$$\sigma_{\hat{\epsilon}}^2 \simeq E\left\{(t-\hat{y})^2\right\} - \sigma_{\hat{y}}^2 \tag{15}$$

From (15), a set of variance squared residuals can be constructed. These residuals can be used to create a new dataset in which an NN model can be trained to estimate values of $\sigma_{\hat{\epsilon}}^2$.

$$r_i^2 = max\left((t_i - \hat{y}_i)^2 - \sigma_{\hat{y}_i}^2, 0\right) \tag{16}$$

The cost function for training the residual model is as follows:

$$C_{BS} = \frac{1}{2}\sum_{i=1}^{n}\left[ln\left(\sigma_{\hat{\epsilon}_i}^2\right) + \frac{r_i^2}{\sigma_{\hat{\epsilon}_i}^2}\right] \tag{17}$$

Once trained, the residual CNN-LSTM provides an estimate of $\sigma_{\hat{\epsilon}_i}^2$. By adding $\sigma_{\hat{y}_i}^2$ to the estimated value of $\sigma_{\hat{\epsilon}_i}^2$ we can obtain an estimate of $\hat{y}_i$, the variance of our prediction. This variance is then used to construct confidence bounds.

The bootstrap method is one of multiple methods that have been used to construct confidence bounds on neural networks. Other methods, such as the Delta method [11] and the Bayesian method [11] require calculating the Hessian matrix, which is too computationally demanding for a dataset of this size. The bootstrap method, while extremely time intensive for training, is a feasible option for constructing confidence bounds. To our knowledge, the development of bootstrapped confidence bounds for Convolutional LSTM networks for single and multi-step predictions is novel to this work.

V. Dataset

In this work, we use data from The Southern California Bight (SCB) ocean forecasting system [18]. This data covers an area of $31.3^o - 43.0^o$ latitude and $232.5^o - 243.0^o$ longitude from March 01, 2014 to April 15, 2016. The time data is discretized to four times per day: 03:00:00, 09:00:00, 15:00:00 and 21:00:00. The data points resolution is 0.03^o in both directions. The data is broken up into smaller 50x50 grid areas, see Fig. 2. These locations were selected due to their proximity to land and similarities in ocean climate. They are not treated as separate locations, but rather ten examples of ocean data.

At this point, there are ten videos comprised of 3104 frames. Videos, 200 frames in length, are extracted from the longer length videos, with 150 frames of overlapping video. This results in 320 videos comprised of 200 frames each. The data is then split into four datasets; Training Set A is 120 videos of 200 frames of 50 by 50 data (120x200x50x50), Training Set B is 80x200x50x50, Validation Set is 80x200x50x50 and the Test Set is 40x200x50x50. Training Set A and B are both used in training neural networks and GP's, while the validation set is used for validating any results during research. Only after all techniques are finalized is the test set used to evaluate results. All results in this work are shown on the test set only.

VI. Results and Discussion

In this section we report on the results achieved in predicting ocean data using GP regression and CNN-LSTM neural networks. We quantify each technique's performance by calculating the average prediction error of future days ocean temperature and the confidence bounds generated for each technique. Additionally, we run simulations of a glider using model estimates to plan paths through an ocean, attempting to monitor areas of the ocean where the temperature of the water falls between 17°C and 17.5°C degrees. Such data would be useful for oceanographers monitoring coastal upwellings or other phenomena [5]. We assume the glider has an accurate position estimate at all times and is unaffected by ocean currents. All predictions are performed on the test set, which has no overlap of data with Training Set A, Training Set B or the Validation Set.

A. Gaussian Process Methods

In this work, we compare our technique to Gaussian Process models of the ocean environment. These predictions have been shown to be effective in modeling environmental phenomena [5].

It is shown in [10] that GP variance fails to predict the uncertainty in ocean processes. Because of this, we also compare our proposed technique to interpolation variance. Interpolation variance provides an alternative measure of uncertainty by incorporating correlations between the data learned in the GP

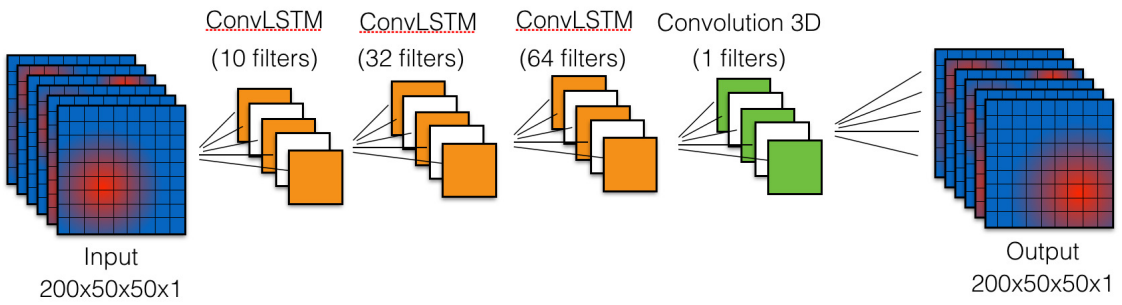

Fig. 1: An illustration of our network structure. A 200 frame times series set of 50x50 images with one channel is used as an input into a 4 layer convolutional LSTM network. The network outputs 200 frames of 50x50 images predicting the value of the next frame. The network is trained through supervised learning for 300 epochs and is used to predict ocean data changes.

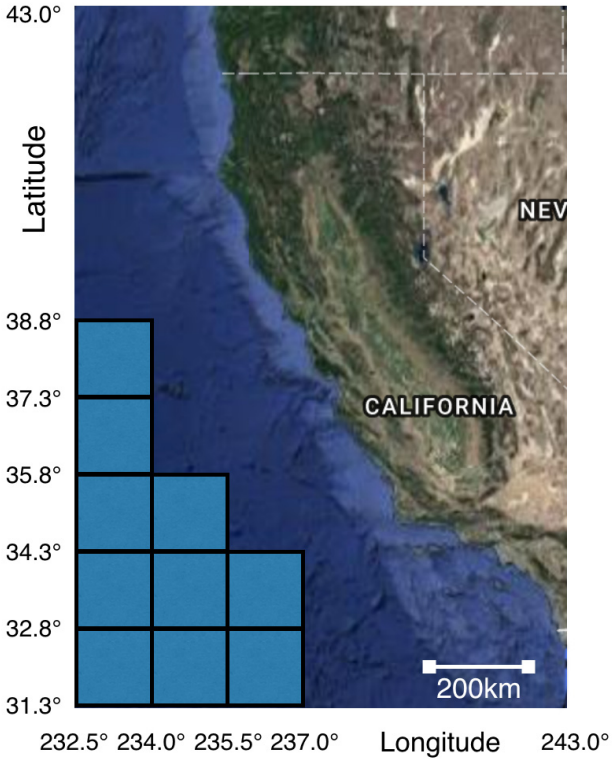

Fig. 2: The data used in this work was pulled from a larger group of data and broken into 50x50 grid squares. The blue squares indicate the relative location of data used. This data covers an area of $31.3^o - 43.0^o$ latitude and $232.5^o - 243.0^o$ longitude from March 01, 2014 to April 15, 2016.

framework with the variability of correlated data to determine the predicted variance. Interpolation variance is defined as:

$$V_{iv}(x_*) = \mathbf{k}_*^T(\mathbf{K} + \sigma_n^2\boldsymbol{I})^{-1} * \sum_{i=1}^{n}(z_i - \mu(x_*))^2 \qquad (18)$$

The GP used in this work is constructed using data from the previous 10 frames of video. The data is sampled at every fifth point across the 50x50 image, resulting in 1000 data points in each GP model trained with a squared-exponential kernel. A new model is constructed for each frame of the video. Each model is trained to predict the difference from the mean temperature value of the previous 10 days. The interpolation variance is constructed using these same sampled data points.

B. Prediction and Confidence Bounds

In this work, we train 20 CNN-LSTM networks, as described above, for 300 epochs on 50 of 80 randomly selected videos in Training Set A. This makes up our ensemble for the bootstrap method. Estimates on Training Set B are then made by all 20 CNN-LSTM networks to obtain a mean and variance used in construction of a set of variance squared residuals. From these residuals, we obtain a new dataset for which a new NN model is trained to estimate $\sigma^2_{\hat{\epsilon}_i}$. This new CNN-LSTM network uses the same architecture as above. This network used the customized error function described in (17) and is trained for 150 epochs.

At this point, 20 CNN-LSTM networks have been trained on sampled Training Set A and 1 CNN-LSTM network has been trained to predict $\sigma^2_{\hat{\epsilon}_i}$. By adding $\sigma^2_{\hat{y}_i}$ to the estimated value of $\sigma^2_{\hat{\epsilon}_i}$ we can obtain an estimate of $\hat{y}_i$, the variance of our estimate. This variance is then used to construct confidence bounds.

To evaluate these techniques, the first metric we look at is the convolutional LSTM and GP's ability to predict multiple values into the future. With the GP, this is simply done by querying the model for the specific location and time. For the NN, this is done by predicting one day in advance (what the model is trained to do) and then appending that estimate onto the time series and querying again. Repeating this process allows us to predict any arbitrary distance into the future. Figure 3 shows the results of querying 10 successive values. Initially, the GP is more accurate, but rapidly loses

accuracy. The NN maintains a much smaller estimate error as the predictions continue.

To evaluate confidence bounds, we use the same metrics as [11]. PICP is a measure of the average width of the confidence bounds. MPIW is the percentage of all true values that fall within the confidence bound estimate and CWC is an overall evaluation of the quality of the confidence bounds (the smaller value representing a higher quality confidence bound). CWC is formally defined below:

$$CWC = MPIW\left(1 + \gamma\left(PICP\right)e^{-\eta(PICP-\mu)}\right) \quad (19)$$

where $\gamma(PICP)$ is given by :

$$\gamma = \begin{cases} 0, & PICP \geq \mu \\ 1, & PICP < \mu \end{cases} \quad (20)$$

where γ and μ are hyper-parameters selected to be the same values as in [11], 50 and .68, respectively. A single standard deviation is used to construct confidence bounds from variance, hence the 0.68 μ value.

Figure 5 shows the average size of the confidence bounds given the 3 techniques. It also shows what percentage of the data falls within the confidence bounds given the prediction interval. Here we see the two GP methods have smaller CB's, but as the prediction interval increases, the CB's coverage shrinks to a level much lower than 68%. This percentage is an important threshold as it is the amount of data that should fall within one standard deviation assuming normal noise. On the other hand, the CNN's confidence bounds are very conservative and grow faster than needed to maintain 68% coverage. To get a general idea of the value of each CB, we can use the CWC score discussed above.

Figure 4 shows the CWC score for the three techniques. Here we see that the GP variance does quite poorly. This is due to the fact that it never achieves 68% coverage, which dramatically hurts its score. The GP interpolation variance does very well for the first few predictions, but as the predictions get farther and farther into the future, it struggles to maintain wide enough bounds. Only the NN keeps wide enough bounds to maintain above a 68% coverage for all prediction intervals. It does this at great cost however, its bounds are significantly wider than either GP method. Overall, using the evaluation method found in [11], we see that the proposed technique of bootstrapped CNN-LSTM neural networks achieve significantly better results then either GP method. This advantage only grows as the the prediction interval increases.

C. Spatio-Temporal Monitoring

To understand the impact of improved environment modeling, we perform a series of simulations of a glider exploring an ocean. Each simulation starts by placing a glider randomly in a 50x50 grid world, no closer than 5 spaces to any edge. The glider receives an estimate from either a GP model or NN model of what the world will look like over the next 10 time steps and makes a plan to maximize its time spent in grid squares where the temperature is between 17°C and

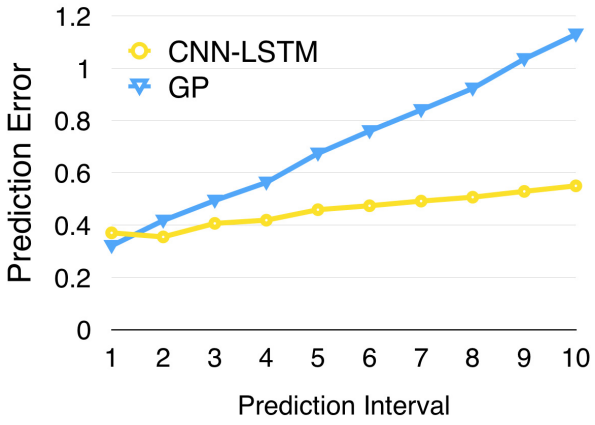

Fig. 3: The average prediction error of the GP and the CNN-LSTM as the prediction interval increases.

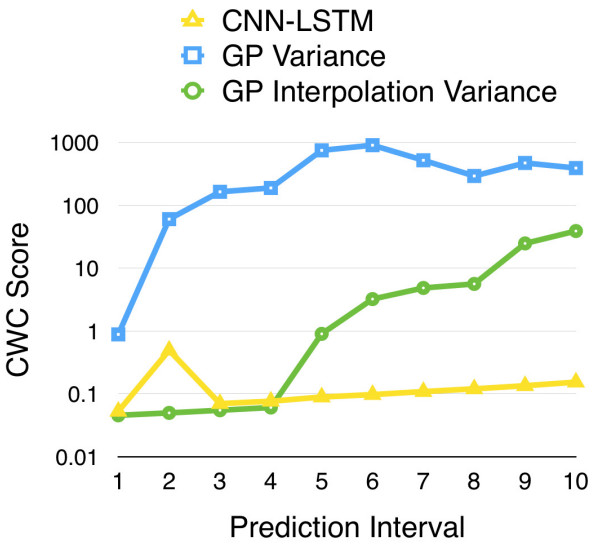

Fig. 4: The CWC score for the three variance measures. CWC is an overall evaluation of the quality of confidence bounds. A smaller value represents a higher quality confidence bound.

17.5°C degrees. Each simulation starts 10 data points into the dataset and ends at datapoint 180. The glider selects its path by performing Monte Carlo Tree Search (MCTS) at each time step. This allows the planner to take advantage of the accurate long-horizon estimates and confidence intervals from the CNN-LSTM network. MCTS searches the estimates generated by the GP, interpolation variance or the CNN-LSTM for the best next move given the model's estimate. We utilize both the mean $u(x)$ and variance $\sigma(x)$ provided by the models by calculating the probability of any particular location falling between the threshold values T_{min} and T_{max}. The probability is calculated as

$$U = \int_{T_{min}}^{T_{max}} P(T)dT \quad (21)$$

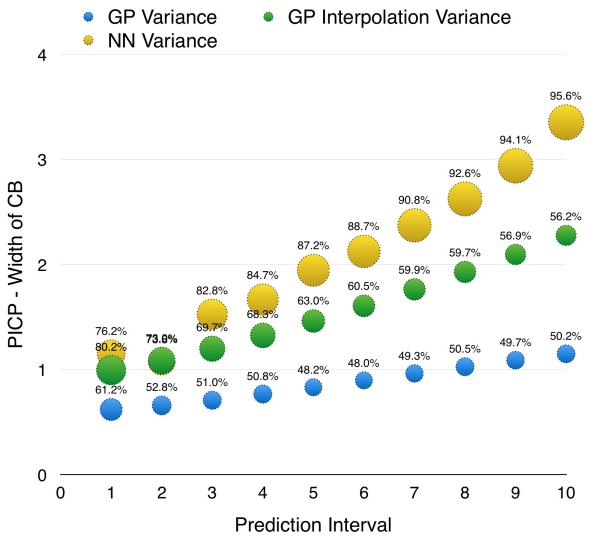

Fig. 5: The average size of the confidence bounds is compared to the percentage of data falling within those bounds, represented by the size of the circle. The GP variance has the thinnest confidence bounds, but also has the least data coverage. The interpolation variance grows the width of the CB but improves coverage. The CNN-LSTM has the widest confidence bounds, but also has the most data coverage, providing the most conservative CBs.

where

$$P = \mathcal{N}(u, \sigma). \tag{22}$$

This probability is used as the reward in the MCTS search, pushing the glider to select paths that are the most likely to result in ocean temperatures between threshold values. A max depth of ten is used in the MCTS search (our models have generated projects of depth ten, hence the hard limit on MCTS depth) and a discount reward structure is used. Five MCTS simulations are run for each example video, with the largest score being selected for comparison.

Figure 6 shows the results for each of the three techniques evaluated on the test set with random starting locations. Results are displayed as a proportion of the score achieved vs the score achieved by having perfect knowledge of the next 10 days. The Gaussian process with standard variance achieves a score 77.2% of perfect knowledge. When the GP variance is replaced with interpolation variance, the proportion increases to 80.7%. Our proposed technique of convolutional LSTM with bootstrapped confidence bounds achieves the best proportion of 83.1%. These results mirror what we see when we look at the raw data. The GP estimate and variance performs poorly at long lookaheads and this performance is reflected in the relatively poor performance in ocean monitoring. Replacing the GP variance with interpolation variance improves performance by increasing the amount of data included within the confidence bounds with minimal increase to the width of the bounds. The CNN-LSTM neural network improves on the long range estimates of ocean temperatures and provides confidence

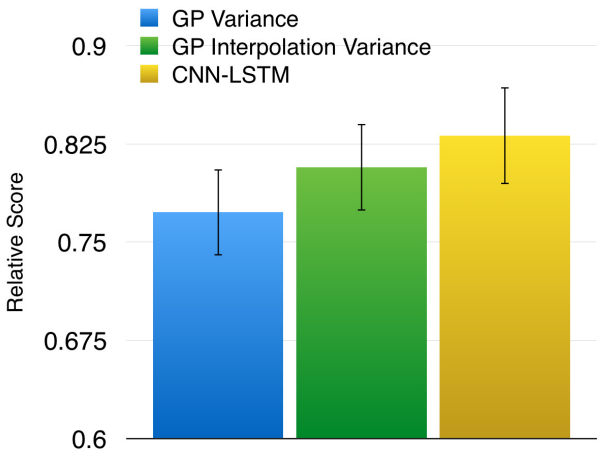

Fig. 6: The result of each technique displayed as a proportion of their score achieved vs the score achieved by having perfect knowledge of future ocean temperatures. Error bars are in SEM. Graph is zoomed in for detail.

bounds that allow for an improvement of 2.4% over the GP with interpolation variance model.

VII. Conclusion and Future Direction

The results presented in this paper demonstrate the promise of using convolutional LSTM neural networks to model spatio-temporal data. This work shows that a deep neural network with bootstrapped confidence bounds outperforms current methods of modeling ocean environments and provides confidence bounds competitive with the state-of-the-art GP techniques. Furthermore, this method is generic and can be applied to any spatio-temporal dataset where estimates and confidence bounds of future events are needed.

Further work is needed in exploring the CNN-LSTM architectures possible for this type of modeling. The number of layers, filters, activation functions and filter size are all parameters in this work and optimizing these could potentially lead to improvements in modeling. The construction of a CNN-LSTM network to estimate $\sigma^2_{\hat{\epsilon}_i}$ also led us to the issue that it only predicts a single prediction interval ahead, which loses accuracy as you predict multiple prediction intervals ahead. The implementation of MCTS could also be extended to include sampled data from distributions. Instead of precomputing all of the mean and variances ahead of running MCTS, each step of MCTS could sample values from a distribution constructed from the mean and variance. This would allow us to capture temporal correlations, utilizing our predictions more effectively.

Lastly, we are interested in combining the predictions of the CNN-LSTM and the MCTS into a end-to-end system. The current technique estimates a 50x50 grid of data in which only a very small portion is used in the MCTS exploration. If an end-to-end CNN-LSTM system was trained to predict spatio-temporal data and estimate MCTS results, a reduced state space may result in improved performance.

References

[1] Jonathan Binney and Gaurav S Sukhatme. Branch and bound for informative path planning. In *IEEE International Conference on Robotics and Automation*, pages 2147–2154, 2012. doi: 10.1109/ICRA.2012.6224902.

[2] Jennifer Casper and Robin R. Murphy. Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 33(3):367–385, 2003.

[3] Hyeong Soo Chang, Michael C Fu, Jiaqiao Hu, and Steven I Marcus. An adaptive sampling algorithm for solving Markov decision processes. *Operations Research*, 53(1):126–139, 2005.

[4] Howie Choset and Philippe Pignon. Coverage path planning: The boustrophedon cellular decomposition. In *Field and service robotics*, pages 203–209. Springer, 1998.

[5] Jnaneshwar Das, Frédéric Py, Harshvardhan Vathsangam, Rishi Graham, Kanna Rajan, and Gaurav S. Sukhatme. Hierarchical probabilistic regression for AUV-based adaptive sampling of marine phenomena. In *IEEE International Conference on Robotics and Automation*, pages 5571–5578. IEEE, 2013.

[6] Jnaneshwar Das, Frédéric Py, Julio BJ Harvey, John P Ryan, Alyssa Gellene, Rishi Graham, David A Caron, Kanna Rajan, and Gaurav S Sukhatme. Data-driven robotic sampling for marine ecosystem monitoring. *The International Journal of Robotics Research*, 34(12): 1435–1452, 2015.

[7] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850v5*, 2013.

[8] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[9] Geoffrey A Hollinger and Gaurav Sukhatme. Sampling-based Motion Planning for Robotic Information Gathering. In *Robotics: Science and Systems*, 2013.

[10] Geoffrey A Hollinger, Arvind A Pereira, and Gaurav S Sukhatme. Learning uncertainty models for reliable operation of autonomous underwater vehicles. In *IEEE International Conference on Robotics and Automation*, pages 5593–5599. IEEE, 2013.

[11] Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F Atiya. Comprehensive review of neural network-based prediction intervals and new advances. *IEEE Transactions on neural networks*, 22(9):1341–1356, 2011.

[12] James J Kuffner and Steven M LaValle. RRT-connect: An efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 995–1001. IEEE, 2000.

[13] M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[14] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. Face recognition: A convolutional neural-network approach. *IEEE Transactions on Neural Networks*, 8(1):98–113, 1997.

[15] Chang-Shing Lee, Mei-Hui Wang, Guillaume Chaslot, Jean-Baptiste Hoock, Arpad Rimmel, Olivier Teytaud, Shang-Rong Tsai, Shun-Chin Hsu, and Tzung-Pei Hong. The computational intelligence of MoGo revealed in Taiwan's computer Go tournaments. *IEEE Transactions on Computational Intelligence and AI in games*, 1(1): 73–89, 2009.

[16] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016.

[17] Sergey Levine, Peter Pastor, Alex Krizhevsky, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *arXiv preprint arXiv:1603.02199*, 2016.

[18] Z. Li. JPL OurOcean portal, 2015. URL http://ourocean.jpl.nasa.gov/.

[19] Roman Marchant, Fabio Ramos, and Scott Sanner. Sequential Bayesian Optimisation for Spatial-Temporal Monitoring. In *The Thirtieth Conference on Uncertainty in Artificial Intelligence*, pages 553–562. AUAI Press, 2014.

[20] Ruben Martinez-Cantin, Nando de Freitas, Arnaud Doucet, and José A Castellanos. Active Policy Learning for Robot Planning and Exploration under Uncertainty. In *Robotics: Science and Systems*, pages 321–328, 2007.

[21] Joseph L. Nguyen, Nicholas R. J. Lawrance, Robert Fitch, and Salah Sukkarieh. Real-time path planning for long-term information gathering with an aerial glider. *Autonomous Robots*, 40(6):1017–1039, 2016.

[22] Tom Pepels, Mark HM Winands, and Marc Lanctot. Real-time monte carlo tree search in Ms Pac-Man. *IEEE Transactions on Computational Intelligence and AI in games*, 6(3):245–257, 2014.

[23] Daniel L Rudnick, Russ E Davis, Charles C Eriksen, David M Fratantoni, and Mary Jane Perry. Underwater gliders for ocean research. *Marine Technology Society Journal*, 38(2):73–84, 2004.

[24] Colin D Ward and Peter I Cowling. Monte Carlo search applied to card selection in Magic: The Gathering. In *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on*, pages 9–16. IEEE, 2009.

[25] Shi Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*, pages 802–810, 2015.