Progress Report

Bardia Mojra

January 29, 2021

Robotic Vision Lab

The University of Texas at Arlington

1 To Do

- Implement basic estimation and generate a baseline for incremental advancement: Currently, I can train a basic model using the randomly generated data. I am currently working on resolving a visualization issue.
- Implement DOPE with added dropout before each layer to estimate variational Bayesian inference.
- Read Data Association and Localization of Classified Objects in Visual SLAM: in progress.
- Work on Pose Estimation Uncertainty paper.
- Implement PoseCNN, DOPE, and BayesOD.
- Look into domain randomization and adaptation techniques.
- Search for recent pose estimation survey papers: Found this paper, [1], on pose estimation.
- As I read more papers, go back and update Pose Estimation Survey paper.
- Finish Tensorflow tutorials, [2]: 4/9 chapters done.
- Learn to implement transfer learning.
- (On pause) Finish Docker tutorials, [3]: 4/18 chapters done.

2 Literature Reviews

2.1 Dropout as a Bayesian Approximation: Representing Model Uncertainty Deep Learning

In this paper [4], the authors introduce a mathematical framework that casts training deep neural nets with dropout layers as approximate Bayesian inference in deep Gaussian processes. They show that dropout layers in deep networks can minimize Kullback-Leibler divergence between an approximate distribution and the posterior of a deep Gaussian process, [5]. This is quite effective since dropout layers preserve Gaussian processes' variational distributions.

To make posterior weight distribution $p(\omega|D)$ intractable, columns are set to zero at random and we end up with $q(\omega)$, an approximate posterior weight distribution. Moreover,these layers are highly multi-modal and induces strong joint correlations over the rows of the network weight matrices W_i.

The objectice KL divergence minimization is computed by approximating each term by Monte Carlo integration with a single sample,$\widehat{\omega}_n \sim q(\omega)$. This allows for unbiased estimation of each posterior modality with $-log\ p(y_n|x_n, \widehat{\omega_n})$ combined with the second summation term they obtained the following:

$$\mathcal{L}_{GP-MC} \propto \frac{1}{N}\sum_{n=1}^{N}\frac{-log\ p(y_n|x_n,\widehat{\omega}_n)}{\tau} + \sum_{l=1}^{L}(\frac{p_i l^2}{2\tau N}\|\mathbf{M}_i\|_2^2 + \frac{l^2}{2\tau N}\|\mathbf{m}_i\|_2^2)$$

Where

$$E(y_n, \widehat{y}(x_n, \widehat{\omega}_n)) = -log\ p(y_n|x_n, \widehat{\omega}_n)/\tau$$

and approximate predictive distribution function represented by:

$$q(y^*|x^*) = \int p(y^*|x^*, \omega)q(\omega)d\omega$$

Which means this Monte Carlo estimation, *MC Dropout*, makes generalized predictions by averaging the weights of the network. This would be similar to T stochastic forward pass through the network and averaging the outputs but rather computationally more efficient.

For regression tasks, predictive log-likelihood estimate is given by:

$$log\ p(y^*|x^*, X, Y) \approx logsumexp(-\frac{1}{2}\tau\|y - \widehat{y}_t\|^2) - log\ T - \frac{1}{2}log\ 2\pi - \frac{1}{2}log\ \tau^{-1}$$

In their experiments, they used a LeNet-5 with dropout layers added. For regression, MC Dropout with .1 or.2 probabilities (resulted in identical uncertainties) were applied before every weight layer and with TanH activation. For classification, they applied a dropout layer before the last fully connected inner-product layer. They use probability of .5 for dropout layers in classification tasks.

3 Progress

Following items are listed in order of priority:

- Object Pose Estimation with Uncertainty: I started a Tensorflow [6] project based on a simple implementation [7]. The code is written using Tensorflow Graphics module which is currently on supported by Google Colab, I am in the process of porting the code so it will use TensorBoard 3D instead. Next, I will expand the 3-layer nextwork to five layers, train, test and compare the results. Later, I will add dropout layers for uncertainty analysis in regression.

  I am currently restructuring the code into modules for model build, train, and testing. Other modules such as feature extraction can also be added to further enhance the performance.

- Implement PoseCNN, DOPE, and BayesOD.

- Look into domain randomization and adaptation techniques.

- Search for recent pose estimation survey papers: Found this paper, [1], on pose estimation.

4 Plans

Following items are listed in order of priority:

- Implement multiple object pose estimation with uncertainty estimation.

- Keep working on Bayesian Pose Estimation paper.

- ARIAC: For now, I will focus on implementing pose estimation and BayesOD implementations.

- Continue on UE4 tutorials.

- Pose Estimation Survey Paper Feedback: On hold, I am working on Bayesian Pose Estimation.

- Project Alpe with Nolan: On pause for right now.

- UR5e: Finish ROS Industrial tutorials.

5 2021 Goals and Target Journals/Conferences

- Submit a paper on pose estimation with uncertainty to ICIRS.
- Get comfortable with TensorFlow and related Python modules.
- Keep writing.

References

[1] G. Du, K. Wang, S. Lian, and K. Zhao, "Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review," *Artificial Intelligence Review*, pp. 1–58, 2020.

[2] B. Planche and E. Andres, "Hands-on computer vision with tensorflow 2," 2019.

[3] G. Schenker, *Learn Docker – Fundamentals of Docker 19.x: Build, test, ship, and run containers with Docker and Kubernetes, 2nd Edition.* Packt Publishing, 2020.

[4] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," 2016.

[5] A. C. Damianou and N. D. Lawrence, "Deep gaussian processes," 2013.

[6] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[7] "graphics/6dof_alignment.ipynb at master · tensorflow/graphics." `https://github.com/tensorflow/graphics/blob/master/tensorflow_graphics/notebooks/6dof_alignment.ipynb`. (Accessed on 01/28/2021).