# Latent Discriminant deterministic Uncertainty
## Supplementary Material

Gianni Franchi[1†], Xuanlong Yu[1,2†], Andrei Bursuc[3], Emanuel Aldea[2], Severine Dubuisson[4], and David Filliat[1]

[1] U2IS, ENSTA Paris, Institut polytechnique de Paris
[2] SATIE, Paris-Saclay University
[3] valeo.ai
[4] CNRS, LIS, Aix Marseille University

## Table of Contents

## A Connection with kernel methods

This section provides a connection between LDU (section 3.2) and kernel ridge regression. Let us consider the generic training of a kernel ridge regressor model [6] $\mathtt{f}$ on the training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, where $n$ is the number of training samples. Let us denote by $k$ the kernel. The Representer Theorem [6] states that the solution to this problem is of the form $\mathtt{f}(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x})$, where $\alpha_i$ are the parameters to optimize.

Therefore, one should evaluate the kernel centered in each training sample. If one relates the proposed LDU model to kernel ridge regression, the main distinction is that we do not evaluate it across the entire training set, but across a smaller set composed of the prototypes. Hence $\mathtt{f}(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i k\left(p_i, h_{\boldsymbol{\omega}}(\mathbf{x}_i)\right)$ with $k(\cdot,\cdot) = \exp(-S_c(\cdot,\cdot))$, which is a composition of kernel positive definite functions. In contrast with DUQ [7] and similarly to SNGP [3] , we approximate the kernel ridge regression based on the prototype set which allows us to simplify the training. However for SNGP, the authors choose to approximate the kernel differently, by relying on the random projection trick [5].

## B Implementation details

### B.1 Classification and semantic segmentation experiments

This section provides the hyper-parameters used in the classification and semantic-segmentation experiments. Our code is implemented in PyTorch [4]. We used

| Hyper-parameter | CIFAR-10 |
|---|---|
| backbone | ResNet18 |
| initial learning rate | 0.1 |
| batch size | 128 |
| lr decay ratio | 0.1 |
| lr decay epochs | [80, 160, 200] |
| number of train epochs | 250 |
| weight decay | 0.0001 |
| cutout | False |
| SyncEnsemble BN | False |

**Table A1. Hyper-parameter configuration used in the classification experiments (§4.1).**

| Hyper-parameter | MUAD | Cityscapes | BDD-Anomaly |
|---|---|---|---|
| Architecture | Deeplab v3+ | Deeplab v3+ | PSPNet |
| backbone | ResNet50 | ResNet50 | ResNet50 |
| output stride | 8 | 8 | None |
| learning rate | 0.1 | 0.1 | 0.02 |
| batch size | 16 | 16 | 4 |
| number of train epochs | 50 | 50 | 30 |
| nb Prototypes | 22 | 22 | 30 |
| weight decay | 0.0001 | 0.0001 | 0.0001 |
| SyncEnsemble BN | False | False | False |
| Cutout | False | False | False |
| random crop of training images | 768 | 768 | None |

**Table A2. Hyper-parameter configuration used in the semantic segmentation experiments (§4.2).**

30 prototypes for BDD-Anomaly and 22 for the other semantic segmentation dataset. $g_\omega^{\text{unc}}$ is for all the semantic segmentation dataset an MLP with one hidden layer, and the number of neurons in the hidden layer is equal to the number of prototypes. For classification, $g_\omega^{\text{unc}}$ is a single fully connected layer. All the parameters are introduced in Table A1 for the classification and Table A2 for the semantic segmentation.

| **Hyper-parameter** | **KITTI** |
|---|---|
| Architecture | BTS [2] |
| backbone | DenseNet161 [1] |
| initial learning rate | 0.0001 |
| batch size | 4 |
| number of train epochs | 50 |
| weight decay | 0.01 |
| random crop of training images | (352, 704) |
| nb Prototypes | 30 |

**Table A3. Hyper-parameter configuration used in the monocular depth estimation experiments (§4.3).**

### B.2 Monocular depth experiments

As mentioned in the main paper, we train all models using the same hyper-parameters as in the original BTS code [2]. We use 30 prototypes in the DM layer for our LDU model. For Single-PU, we duplicate the top layer and double the number of output channels of the pre-logit layer. For the last layer, we have two one-channel-map outputs: one for depth estimation, one for uncertainty estimation. For the Deep Ensembles baseline, we train Single-PU models. We provide all our hyper-parameters in Table A3. We will make the code publicly available after the anonymity period.

## C Ablation study

In this section, we provide various ablation studies on evaluating the sensitivity of hyper-parameter choices for classification and regression tasks.

In Table A4 and Table A6 we report the results for different values of $\lambda$, the weight of the extra losses for classification on CIRFAR-10 and for monocular depth estimation on KITTI respectively. The performance across metrics is relatively stable with respect to the choice of $\lambda$ with a good compromise at 0.1.

Concerning the influence of prototype number, according to Table 2 in the main paper and Table A6, the predictive performance is higher with more prototypes, while fewer prototypes make lighter and faster models.

We also study the impact of different losses in the classification task in Table A5. The three losses bring consistent improvements individually and together.

## D Training dynamics

In this section we illustrate the training curves when we train our models with/without applying LDU modifications. We take our regression experiment as an example,

| $\lambda$ | Acc ↑ | AUC ↑ | AUPR ↑ | ECE ↓ |
|---|---|---|---|---|
| 0.01 | 87.93 | 0.8425 | 0.9079 | 0.5319 |
| 0.1 | 87.95 | **0.8721** | **0.9147** | **0.4933** |
| 0.5 | 87.99 | 0.8526 | 0.9109 | 0.5184 |
| 1.0 | 87.88 | 0.8399 | 0.9035 | 0.5005 |
| 2.0 | **88.07** | 0.8339 | 0.9008 | 0.4954 |

**Table A4. Ablation studies for image classification on CIFAR-10.** Sensitivity of $\lambda$ values.

| $\mathcal{L}^{\text{Unc}}$ | $\mathcal{L}^{\text{Entrop}}$ | $\mathcal{L}^{\text{Dis}}$ | Acc ↑ | AUC ↑ | AUPR ↑ | ECE ↓ |
|---|---|---|---|---|---|---|
| ✓ | | | 87.69 | 0.8195 | 0.8600 | 0.5256 |
| ✓ | ✓ | | 87.87 | 0.8362 | 0.9090 | 0.5386 |
| ✓ | | ✓ | **88.04** | 0.8613 | 0.8897 | 0.4973 |
| ✓ | ✓ | ✓ | 87.95 | **0.8721** | **0.9147** | **0.4933** |

**Table A5. Ablation studies for image classification on CIFAR-10.** Impact of proposed losses ($\lambda$=0.1).

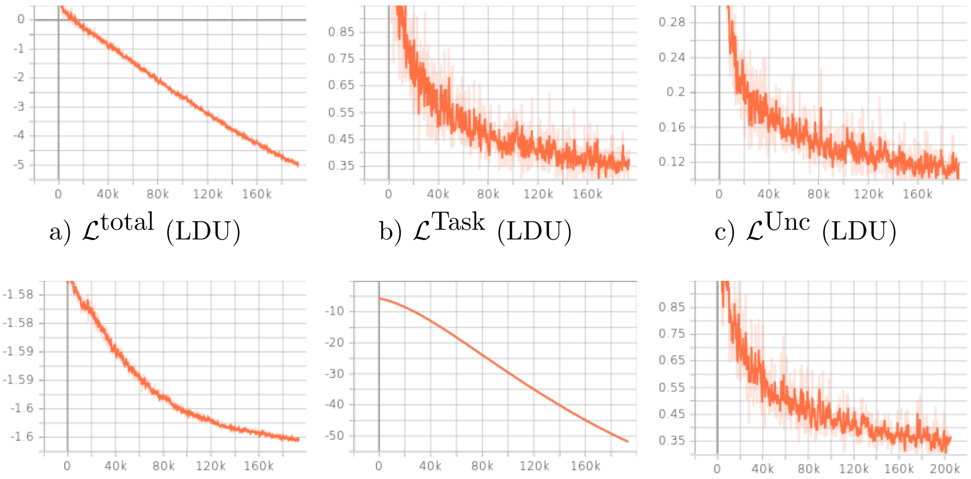
and plot the training curves in Fig. A1. Compared to the original setting, the inserted DM layers and additional losses did not affect the stability of training, all losses decrease smoothly.

## E More visualizations

In Fig. A2 we present qualitative results depicting uncertainty for monocular depth estimation. We show side-by-side depth predictions and uncertainty maps generated by Single-PU, Deep Ensembles and LDU respectively. We observe that for the areas with valid ground truth, all uncertainty estimation strategies highlight the edges of the objects, where the aleatoric uncertainty is frequently prominent. Concerning the areas without valid ground truth, Deep Ensembles does a better job than Single-PU in highlighting them since it can capture more epistemic uncertainty due to the ensembling of multiple predictions from the individual models. Our proposed LDU highlights even better some distant areas, especially the upper part of the image where LiDAR beams do not go. We consider that this result stems from LDU regarding this region as OOD regions after training on the entire dataset.

| #p | λ | d1↑ | d2↑ | d3↑ | Abs Rel↓ | Sq Rel↓ | RMSE↓ | RMSE log↓ | log10↓ | AUSE RMSE↓ | AUSE Absrel↓ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 30 | 0.01 | **0.957** | **0.993** | 0.998 | **0.061** | **0.246** | 2.768 | **0.097** | **0.027** | 0.12 | 0.29 |
| | 0.1 | 0.955 | 0.992 | 0.998 | **0.061** | 0.248 | **2.757** | **0.097** | **0.027** | 0.09 | 0.26 |
| | 0.5 | 0.952 | **0.993** | 0.998 | 0.064 | 0.256 | 2.789 | 0.100 | 0.028 | 0.09 | 0.28 |
| | 1.0 | 0.952 | 0.992 | 0.998 | 0.064 | 0.257 | 2.767 | 0.099 | 0.028 | **0.08** | 0.23 |
| 5 | 0.01 | 0.953 | **0.993** | 0.998 | 0.064 | 0.264 | 2.777 | 0.099 | 0.028 | 0.15 | 0.39 |
| | 0.1 | 0.953 | 0.992 | 0.998 | 0.064 | 0.256 | 2.773 | 0.100 | 0.028 | 0.12 | 0.33 |
| | 0.5 | 0.953 | **0.993** | 0.998 | 0.063 | 0.253 | 2.776 | 0.099 | 0.028 | 0.09 | 0.26 |
| | 1.0 | 0.954 | **0.993** | 0.998 | 0.063 | 0.253 | 2.768 | 0.098 | **0.027** | **0.08** | **0.21** |
| 15 | 0.1 | 0.954 | **0.993** | 0.998 | 0.062 | 0.249 | 2.769 | 0.098 | **0.027** | 0.10 | 0.28 |

**Table A6. Ablation studies for monocular depth estimation on KITTI.** Sensitivity of $\lambda$ and the number of prototypes.



a) $\mathcal{L}^{\text{total}}$ (LDU) b) $\mathcal{L}^{\text{Task}}$ (LDU) c) $\mathcal{L}^{\text{Unc}}$ (LDU)

d) $\mathcal{L}^{\text{Entrop}}$ (LDU) e) $\mathcal{L}^{\text{Dis}}$ (LDU) f) $\mathcal{L}^{\text{Task}}$ (original)

**Fig. A1.** Illustrations on training curves for different losses. Figure a) - e): training curves for the model with LDU modifications; Figure f): training curve for the original model. The $\mathcal{L}^{\text{Task}}$ is silog loss for depth regression as defined in BTS [2].

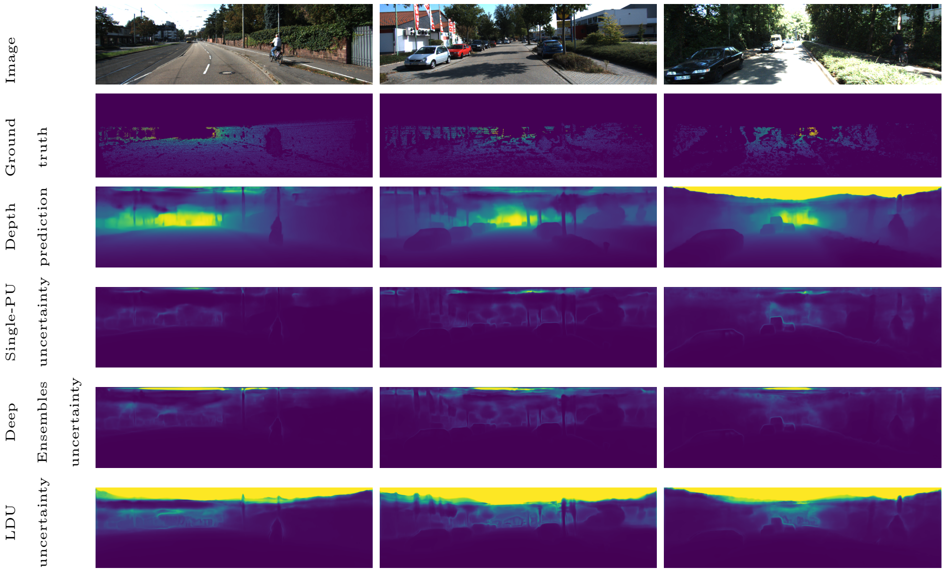


**Fig. A2.** Illustration of different uncertainty maps (LDU, Deep Ensembles, Single-PU) on KITTI images for the monocular depth estimation task (§4.3). For both depth and uncertainty maps, the brighter the color is, the bigger the value the pixel has.

## References

1. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR (2017) 3
2. Lee, J.H., Han, M.K., Ko, D.W., Suh, I.H.: From big to small: Multi-scale local planar guidance for monocular depth estimation. arXiv preprint arXiv:1907.10326 (2019) 3, 5
3. Liu, J.Z., Lin, Z., Padhy, S., Tran, D., Bedrax-Weiss, T., Lakshminarayanan, B.: Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. In: NeurIPS (2020) 1
4. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019) 1
5. Rahimi, A., Recht, B., et al.: Random features for large-scale kernel machines. In: NeurIPS (2007) 1
6. Smola, A.J., Schölkopf, B.: Learning with kernels. Citeseer (1998) 1
7. Van Amersfoort, J., Smith, L., Teh, Y.W., Gal, Y.: Uncertainty estimation using a single deep deterministic neural network. In: ICML (2020) 1