CS 4V98: Final Report

Jerry Xu *

5/7/2021

1 CamVid Dataset

The CamVid dataset [2] consists of 701 images of roadside scenes. The dataset is split into a training set of 367 images, a validation set of 101 images, and a test of 233 images. Each image has size $(3 \times 360 \times 480)$, where 360 is the height of the image in pixels, 480 is the width of the image in pixels, and 3 is the number of *channels* in the image. Each channel corresponds to one of red, green, or blue. The color of an arbitrary pixel is given by the tuple $color = (r, g, b)$ where r, g, and b are in the interval $[0, 255]$. There are 12 classes in the dataset: Sky, Building, Column-Pole, Road, Sidewalk, Tree, Sign-Symbol, Fence, Car, Pedestrian, Bicyclist, and Unlabelled.

(a) Original Image

(b) Labelled Image

Figure 1: An image in the CamVid dataset

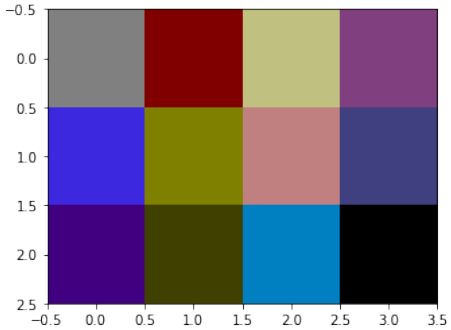


Figure 2: Colors of the 12 classes. Top Row: Sky, Building, Column-Pole, Road. Middle Row: Sidewalk, Tree, Sign-Symbol, Fence. Bottom Row: Car, Pedestrian, Bicyclist, Unlabelled.

*Mentors: Professor Feng Chen and Xujiang Zhao

2 Semantic Segmentation and the One Hundred Layers Tiramisu

The idea behind semantic segmentation is recognizing and understanding what is in an image at a pixel level. Semantic segmentation assigns a class to each pixel in an image, but does not distinguish between separate instances of objects in the same class [4]. One of the most important applications of semantic segmentation is autonomous vehicles, as autonomous vehicles need tools to understand their environment so that they can safely integrate into our existing roads [8].

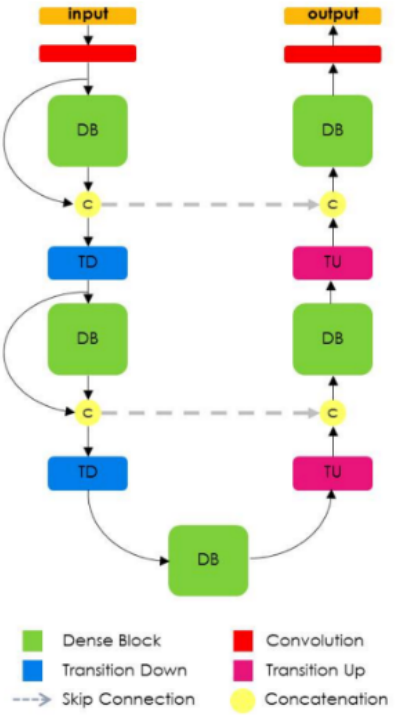
The model I used to semantically segment the images in the CamVid dataset is Tiramisu103 [7], which has 103 convolutional layers. The machine learning frameworks used to implement Tiramisu103 in [7] have fallen out of favor, so I modified the PyTorch Tiramisu103 implementation found in [1]. Tiramisu103 combines two different Convolutional Neural Network (CNN) architectures: UNet [12] and DenseNet [6]. Tiramisu103 has the overall shape of the former and the DenseBlocks of the latter. When given an input of shape $(3 \times 360 \times 480)$, Tiramisu103 produces an output of shape $(12 \times 360 \times 480)$ as there are 12 classes in the CamVid dataset. The building blocks and architecture of the Tiramisu103 model are shown below:

Layer
Batch Normalization
ReLU
3×3 Convolution
Dropout $p = 0.2$

Transition Down (TD)
Batch Normalization
ReLU
1×1 Convolution
Dropout $p = 0.2$
2×2 Max Pooling

Transition Up (TU)
3×3 Transposed Convolution $stride = 2$

Figure 3: Building blocks of fully convolutional DenseNets [7].



Architecture
Input, $m = 3$
3×3 Convolution, $m = 48$
DB (4 layers) + TD, $m = 112$
DB (5 layers) + TD, $m = 192$
DB (7 layers) + TD, $m = 304$
DB (10 layers) + TD, $m = 464$
DB (12 layers) + TD, $m = 656$
DB (15 layers), $m = 896$
TU + DB (12 layers), $m = 1088$
TU + DB (10 layers), $m = 816$
TU + DB (7 layers), $m = 578$
TU + DB (5 layers), $m = 384$
TU + DB (4 layers), $m = 256$
1×1 Convolution, $m = c$
Softmax

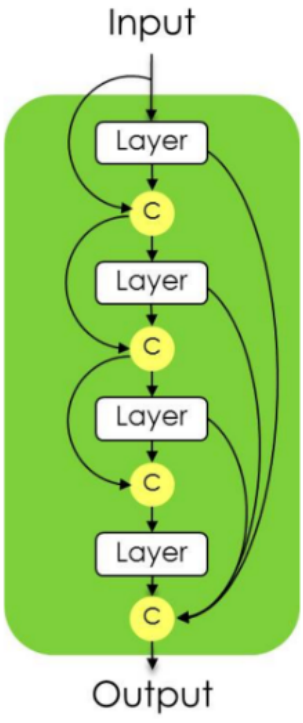


Figure 4: Overall Tiramisu architecture (left and middle) and 4-Layer Dense Block architecture (right) [7].

3 Types of Uncertainty

There are five types of uncertainty that I aimed to quantify in the semantically segmented images. Aleatoric uncertainty is caused by statistical randomness, epistemic uncertainty is caused by limited knowledge or ignorance in the collected data, entropy is the sum of aleatoric uncertainty and epistemic uncertainty, vacuity is uncertainty caused by a lack of evidence, and dissonance is uncertainty caused by conflicting evidence [5].

3.1 Bayesian Neural Networks

Standard deep neural networks have a fixed set of weights and biases associated with them. Thus, a DNN produces only one output when given an input. In a Bayesian neural network, each weight and bias in the network has a probability distribution associated with it. Thus, a BNN produces a range of outputs when given an input. However, applying the Bayesian framework to a DNN such as Tiramisu103 is computationally expensive. To approximate a BNN with a DNN, dropout layers are used. A dropout layer sets a neuron in a DNN to zero with probability p. They have been shown to help prevent a DNN from over-fitting on the training data [13]. In Tiramisu103, the dense block layers and transition down layers in Figure 3 have dropout layers where $p = 0.2$. The DNN is trained with the dropout layers on. Normally, dropout layers are turned off during testing, but that is not the case here. Having the dropout layers on during testing simulates each weight and bias having its own probability distribution. A given input is passed through the DNN T times (in my case, $T = 5$), and the average prediction is calculated by averaging the T individual predictions [9].

3.2 Entropy, Aleatoric Uncertainty, and Epistemic Uncertainty

As aleatoric uncertainty is caused by statistical randomness, it cannot be reduced with more training data. However, epistemic uncertainty can be reduced with more training data [9]. Entropy is defined as the sum of aleatoric uncertainty and epistemic uncertainty. The equation for calculating the entropy H is shown below, where $\hat{x}$ is a test input, $\hat{y}$ its class label, and k is the number of classes [9]:

$$\begin{aligned} H(\hat{y} \mid \hat{x}) &= AU(\hat{y} \mid \hat{x}) + EU(\hat{y} \mid \hat{x}) \\ &= -\sum_{k} p(\hat{y} = k \mid \hat{x}) \times \log p(\hat{y} = k \mid \hat{x}) \end{aligned}$$

The equation for calculating aleatoric uncertainty is shown below, where T is defined in the previous subsection, and w_i is the weights associated with pass i, $1 \leq i \leq T$ through a DNN [9]:

$$AU(\hat{y} \mid \hat{x}) = -\frac{1}{T} \sum_{k,i} p(\hat{y} = k \mid \hat{x}, w_i) \times \log p(\hat{y} = k \mid \hat{x}, w_i)$$

Epistemic uncertainty is the difference between entropy and aleatoric uncertainty [9]:

$$EU(\hat{y} \mid \hat{x}) = H(\hat{y} \mid \hat{x}) - AU(\hat{y} \mid \hat{x})$$

3.3 Vacuity and Dissonance

The ELU (exponential linear unit) function is defined as

$$ELU(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha e^x - 1 & \text{if } x \leq 0 \end{cases}$$

By default, α is set to 1 [3]. According to Xujiang, a DNN used to calculate vacuity and dissonance should have an ELU as the final layer. Tiramisu103 produces an output of shape $(12 \times 360 \times 480)$ as there are 12 classes in the CamVid dataset. The vacuity (uncertainty caused by a lack of evidence) of a prediction $X = \{x_1, \ldots, x_n\}$ is given by the following equation, where n is the number of classes [14]:

$$Vac(X) = \frac{n}{\sum_{k=1}^{n} x_k + 2}$$

The dissonance (uncertainty caused by conflicting evidence) of a prediction $X = \{x_1, \ldots, x_n\}$, where $x_1, \ldots, x_n$ are nonzero, is given by the following equation, where n is the number of classes [14]:

$$Diss(X) = \sum_{i=1}^{n} \frac{x_i \sum_{j \neq i} x_j Bal(x_i, x_j)}{\sum_{j \neq i} x_j}$$

where

$$Bal(x_i, x_j) = 1 - \frac{|x_i - x_j|}{x_i + x_j}$$

4 Metrics

4.1 Area Under Receiver Operating Characteristic (AUROC)

The ROC is a plot of the true positive rate against the false positive rate, and the AUROC is the area under the TP-FP curve. An AUROC value of 0.5 indicates that the classifier is no better than a coin flip, while an AUROC value of 1 indicates that the classifier is perfect [11].

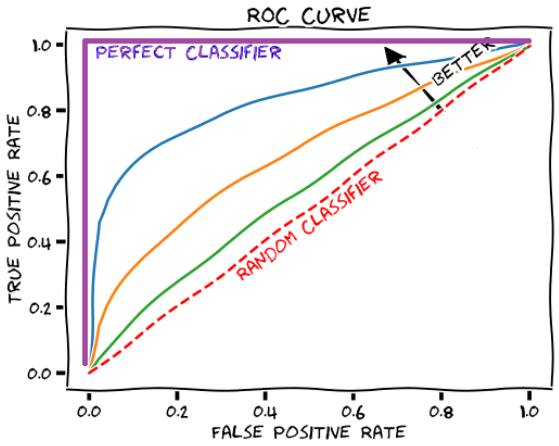


Figure 5: An example of an AUROC curve. [11]

4.2 Area Under Precision/Recall (AUPR)

Precision is defined as $P = \frac{TP}{TP+FP}$ and Recall is defined as $R = \frac{TP}{TP+FN}$. In short, precision answers the question "How many selected items are relevant" and recall answers the question "How many relevant items are selected." AUPR is the area under the plot of precision against recall [10].

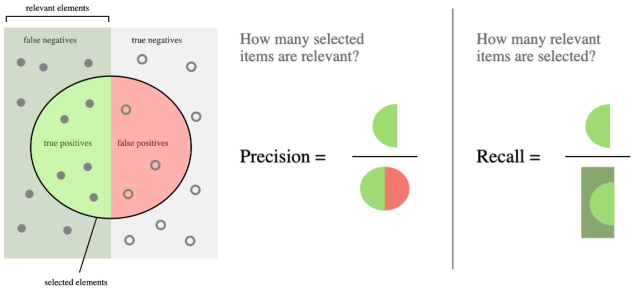


Figure 6: An example of Precision and Recall. [10]

5 Results

5.1 Visualizations

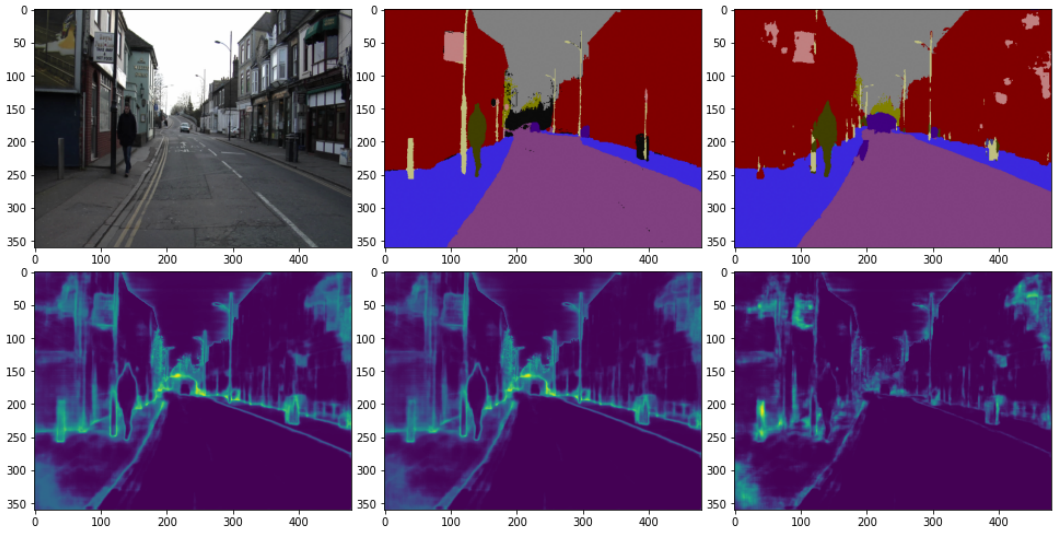


Figure 7: Top Row: original image, ground truth, model's prediction. Bottom Row (brighter color = higher uncertainty): Entropy, Aleatoric Uncertainty, Epistemic Uncertainty.

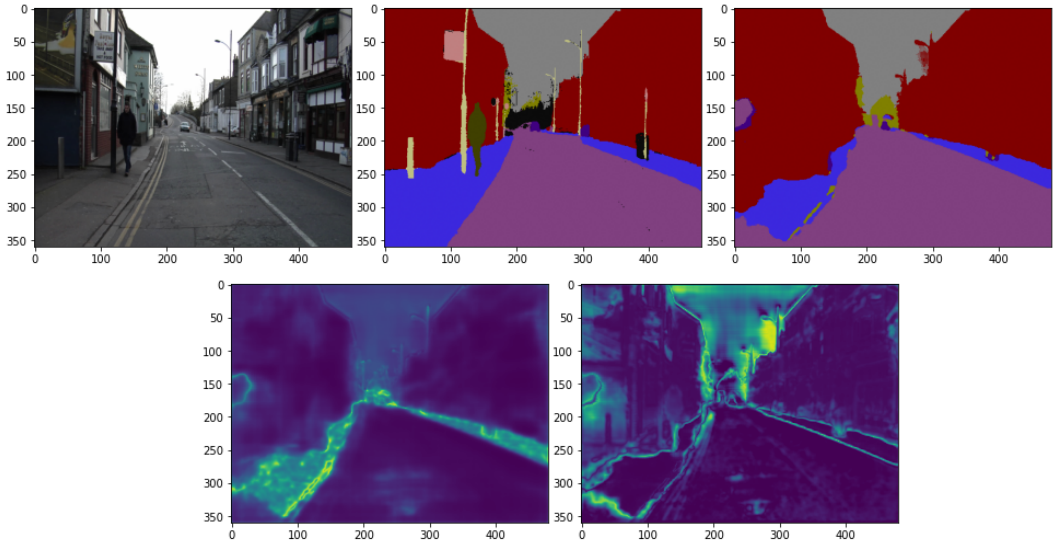


Figure 8: Top Row: original image, ground truth, model's prediction. Bottom Row (brighter color = higher uncertainty): Vacuity, Dissonance.

5.2 Observations

One instance of Tiramisu103 was used to calculate Entropy, Aleatoric Uncertainty, and Epistemic Uncertainty. A separate instance of Tiramisu103 was used to calculate Vacuity and Dissonance as a different loss function is needed. While the Tiramisu103 instance that was used to calculate Entropy, Aleatoric Uncertainty, and Epistemic Uncertainty produced a prediction that is highly similar to the ground truth, the

Tiramisu103 instance that was used to calculate Vacuity and Dissonance produced an output that is rather far from the ground truth. In some cases (see Figure 9), the output was very far from the ground truth.

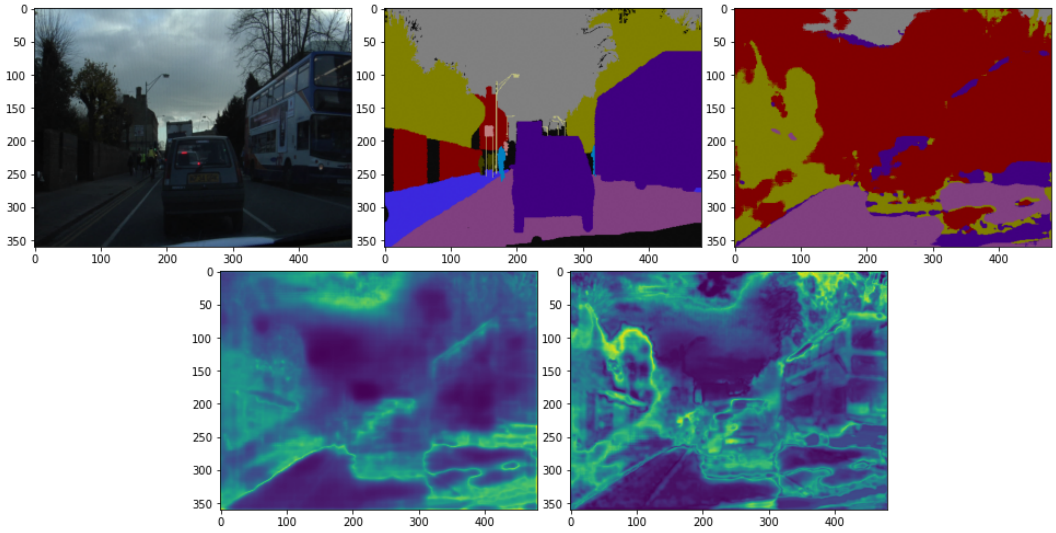


Figure 9: Top Row: original image, ground truth, model's prediction. Bottom Row (brighter color = higher uncertainty): Vacuity, Dissonance.

5.3 Statistics

Figure 7		Figure 8		Figure 9	
Accuracy	0.940	Accuracy	0.866	Accuracy	0.286
Entropy AUROC	0.922	Vacuity AUROC	0.765	Vacuity AUROC	0.443
Entropy AUPR	0.994	Vacuity AUPR	0.955	Vacuity AUPR	0.264
Aleatoric AUROC	0.923	Dissonance AUROC	0.763	Dissonance AUROC	0.589
Aleatoric AUPR	0.994	Dissonance AUPR	0.953	Dissonance AUPR	0.412
Epistemic AUROC	0.879				
Epistemic AUPR	0.991				

References

[1] Brendan Fortuner, Jeremy Howard, Matt Kleinsmith. *PyTorch Tiramisu*. 2018. URL: https://github.com/bfortuner/pytorch_tiramisu.

[2] Gabriel J. Brostow, Julien Fauqueur, and Roberto Cipolla. "Semantic object classes in video: A high-definition ground truth database". In: *Pattern Recogn. Lett.* 30 (2 Jan. 2009), pp. 88–97. ISSN: 0167-8655. DOI: 10.1016/j.patrec.2008.04.005. URL: http://portal.acm.org/citation.cfm?id=1464534.1465403.

[3] *ELU*. 2019. URL: https://pytorch.org/docs/stable/generated/torch.nn.ELU.html.

[4] Fei-Fei Li, Justin Johnson, and Serena Yeung. *CS 231n Lecture 11: Detection and Segmentation*. 2017. URL: http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf.

[5] Yibo Hu et al. "Multidimensional Uncertainty-Aware Evidential Neural Networks". In: *CoRR* abs/2012.13676 (2020). arXiv: 2012.13676. URL: https://arxiv.org/abs/2012.13676.

[6] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. "Densely Connected Convolutional Networks". In: *CoRR* abs/1608.06993 (2016). arXiv: 1608.06993. URL: http://arxiv.org/abs/1608.06993.

[7] Simon Jégou et al. "The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation". In: *CoRR* abs/1611.09326 (2016). arXiv: 1611.09326. URL: http://arxiv.org/abs/1611.09326.

[8] Jeremy Jordan. *An overview of semantic image segmentation*. 2018. URL: https://www.jeremyjordan.me/semantic-segmentation/.

[9] Buu Phan et al. "Bayesian Uncertainty Quantification with Synthetic Data". In: *Computer Safety, Reliability, and Security*. Ed. by Alexander Romanovsky et al. Cham: Springer International Publishing, 2019, pp. 378–390.

[10] *Precision and recall*. URL: https://en.wikipedia.org/wiki/Precision_and_recall.

[11] Rachel Draelos. *Measuring Performance: AUC (AUROC)*. 2019. URL: https://glassboxmedicine.com/2019/02/23/measuring-performance-auc-auroc/.

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597 (2015). arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

[13] Nitish Srivastava et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

[14] Xujiang Zhao et al. "Uncertainty Aware Semi-Supervised Learning on Graph Data". In: *CoRR* abs/2010.12783 (2020). arXiv: 2010.12783. URL: https://arxiv.org/abs/2010.12783.