Evaluating Bayesian Deep Learning Methods for Semantic Segmentation

Jishnu Mukhoti
University of Oxford
jishnu.mukhoti@cs.ox.ac.uk

Yarin Gal
University of Oxford
yarin@cs.ox.ac.uk

Abstract

Deep learning has been revolutionary for computer vision and semantic segmentation in particular, with Bayesian Deep Learning (BDL) used to obtain uncertainty maps from deep models when predicting semantic classes. This information is critical when using semantic segmentation for autonomous driving for example. Standard semantic segmentation systems have well-established evaluation metrics. However, with BDL's rising popularity in computer vision we require new metrics to evaluate whether a BDL method produces better uncertainty estimates than another method. In this work we propose three such metrics to evaluate BDL models designed specifically for the task of semantic segmentation. We modify DeepLab-v3+, one of the state-of-the-art deep neural networks, and create its Bayesian counterpart using MC dropout and Concrete dropout as inference techniques. We then compare and test these two inference techniques on the well-known Cityscapes dataset using our suggested metrics. Our results provide new benchmarks for researchers to compare and evaluate their improved uncertainty quantification in pursuit of safer semantic segmentation.

1. Introduction

Deep learning techniques have had tremendous success in quite a few fields of machine learning including computer vision [36, 60, 18, 31], natural language processing [25, 58, 48, 49, 38], speech recognition [50, 16, 28, 26, 3], bioinformatics [40, 62, 2] and others. However, most deep learning models produce point-estimates as outputs and hence we do not gain any knowledge about the confidence of the model in its predictions. With the increasing use of AI systems in real-life scenarios like autonomous driving [41, 30, 7, 39, 56] and medical diagnosis [17, 59, 54, 44], there are many cases in which additional knowledge about the model's confidence, i.e. capturing whether the model is essentially 'guessing at random', becomes not only useful but essential [4].

The development of new computationally light-weight, scalable methods of performing approximate Bayesian inference in deep neural networks has enabled these models to estimate their uncertainty in addition to making predictions [21, 22, 37, 6, 47, 8]. However, as we do not have ground truth uncertainty values, we cannot use the conventional methods of evaluation to compare and benchmark these models. Furthermore, the metrics designed for evaluating model performance are often task dependent. For instance, the intersection-over-union (IOU) [46] metric is heavily used in computer vision problems like object detection and semantic segmentation. Extending on these ideas, in this work, we propose new specialised metrics to evaluate Bayesian models designed for the task of semantic segmentation.

Semantic segmentation [46] is a difficult problem in computer vision which requires pixel-level understanding of an image. A Bayesian model for semantic segmentation will not only produce predictions for each pixel but also generate pixel-wise uncertainty estimates. As mentioned above, evaluating such a Bayesian model is a challenging task because unlike predictions, we do not have a strong definition of what a good uncertainty estimate is. Hence, we have to judge the quality of the model uncertainty based on how accurate the model is for the same input. We require metrics which look at both the model predictions and uncertainties and take into account general desiderata about when a model should be uncertain about its predictions. In particular, we use the following two intuitive desiderata:

1. **Desideratum 1:** *if a model is confident about its prediction, it should be accurate on the same.*
2. **Desideratum 2:** *if a model is not confident about its prediction, it may or may not be accurate.*

This hints at an inverse relation between model accuracy and uncertainty, a property which we exploit when designing metrics for performance evaluation.

There has been a lot of research on deep architectures for semantic segmentation [46, 55, 5, 9, 11, 10, 13, 66]. In this work, we choose DeepLab-v3+ [13], one of the state-of-the-art neural networks for this purpose and create two probabilistic versions of it using dropout based approximate inference techniques: MC dropout [21] and Concrete dropout

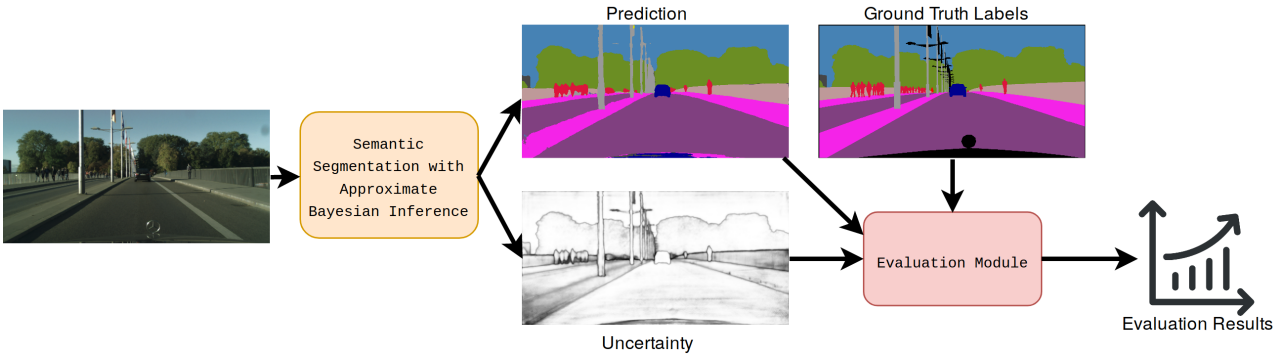


Figure 1. High level overview of the system proposed in this work. The input is first passed through a Bayesian neural network which produces pixel-wise predictions as well as pixel-wise uncertainty estimates. The ground truth labels, predictions and uncertainties are then sent to the performance evaluation module which returns the values of the metrics designed for evaluating the model.

[22]. We train these models on the well-known Cityscapes dataset [15] which contains many images of urban street scenes. Finally, we evaluate and compare the trained models using the metrics which we propose in this work. In Figure 1, we present a high level overview of the evaluation system which we implement.

In a nutshell, the main contributions of this paper are listed as follows:

1. We propose three novel metrics which can be used to evaluate Bayesian models trained for the task of semantic segmentation.
2. We create two probabilistic versions of the DeepLab-v3+ [13] network, which can produce pixel-wise uncertainty estimates in addition to semantic segmentation results.
3. Finally, we evaluate the MC dropout [21] and Concrete dropout [22] inference techniques using the metrics mentioned in the above point, thereby laying down benchmarks against which other models can be compared.

2. Related Work

In this section, we discuss some of the recent works on semantic segmentation as well as those on approximate inference in Bayesian Deep Learning.

2.1. Semantic Segmentation

The work by Long et al. [46] was the first of its kind where a convolutional neural network without any fully connected layers was trained in an end-to-end manner directly mapping images to their corresponding segmentation results. This enabled the network to segment images of varying sizes. However, the fully convolutional networks still suffered due to the presence of pooling layers which ignore the positional information of objects in an attempt to reduce the dimensions of feature maps.

In order to get around this issue, researchers have followed two primary threads of thought: the encoder-decoder architecture [55, 5, 32] and the dilated/atrous convolutions [64, 9, 11, 10]. The encoder-decoder networks first reduce the spatial dimensions of the feature maps with repeated applications of convolution and pooling layers in the encoder module. Next, in the decoder module, the spatial dimensions are gradually recovered using de-convolution and upsampling layers. In order to have sharper segmentation results, skip connections are often introduced between the encoder and decoder modules. Few popular works in this category include U-Net [55], SegNet [5] and RefineNet [42].

The second class of architectures use dilated or atrous convolutions [64] to have a larger field of view over the input feature maps without a decrease in spatial dimensions. One of the most popular set of deep neural networks which follow this policy is DeepLab [9, 11, 10, 13]. In this work, we adopt a combination of both policies and use DeepLab-v3+ [13] as the base network for semantic segmentation. DeepLab-v3+ uses atrous convolution layers as well as a simple decoder module to have fine-grained segmentation outputs.

2.2. Approximate Inference in Deep Neural Nets

The idea behind Bayesian modelling is to find the probability of each set of model parameters given a dataset. In order to do so, an initial distribution known as the *prior* is assumed over the model parameters. Next, with the input of data, this distribution is updated to capture parameters which are more likely to have generated the dataset. This update is done by applying Bayes' theorem. Once the entire dataset has been processed, the distribution over the model parameters, known as the *posterior*, captures the updated belief about the optimal set of parameters to represent the data.

However, getting the posterior for large neural networks

is computationally intractable. Hence, many methods to approximate the posterior have been proposed. One such class of methods include Markov Chain Monte Carlo (MCMC) techniques [51, 63, 1]. Another popular set of techniques uses variational inference [24, 6, 21, 22, 35] where the posterior is approximated using a variational distribution. In this work, we have used MC dropout [21] and Concrete dropout [22] as methods of approximate inference. Both these methods are based on the variational inference approach and are standard well-performing techniques which are easy to implement in deep neural networks.

Finally, we conclude this section with a discussion of Bayesian SegNet [32], which is one of several works to have applied approximate Bayesian inference in the context of semantic segmentation. The authors of [32] modify the SegNet architecture [5] using MC dropout [21] to obtain uncertainties in addition to segmentation results. Furthermore, they present a few accuracy-vs-uncertainty plots in their work, which are good sanity checks for a Bayesian neural network. These sanity checks are qualitative though, and do not allow us to compare and choose between BDL models for semantic segmentation. It is precisely this gap that we fill by developing *quantitative* measures as well.

3. Bayesian DeepLab

In this section we describe the variant of the DeepLab-v3+ [13] network architecture which we have implemented in this work. We begin with a brief review of the architectural features of DeepLab-v3+ and then go on to describe the modifications made to the same to perform approximate inference.

3.1. Brief review of DeepLab-v3+

DeepLab-v3+ is one of the state-of-the-art deep neural networks designed for the problem of semantic segmentation. There have been multiple versions of DeepLab, namely DeepLab-v1 and v2 [9, 10], DeepLab-v3 [11] and DeepLab-v3+ [13], over the years. However, all these versions possess certain common architectural traits. Firstly, they propose atrous or dilated convolutions [29, 23, 52] as a way to widen the field of view over the input feature maps without increasing the number of parameters or using pooling layers. Secondly, they deal with the problem of objects present at different scales in the image using methods like image pyramid [19, 43, 53, 12], atrous spatial pyramid pooling (ASPP) [10, 66], cascaded atrous modules [11, 45, 65] and encoder-decoder architectures [5, 55]. Thirdly, even though there are no pooling layers in the network, due to the presence of multiple convolution layers with strides of 1 or more, the resulting output dimensions are reduced. In order to regain the original dimensions, the output is passed through a fully connected CRF [9, 10] or resized using bilinear interpolation [11] or passed through a decoder with learnable parameters [13].

Finally, the above-mentioned architectural features of DeepLab can be applied to any base network as long as it is fully convolutional. Some of the popular deep CNNs which have been used for DeepLab are VGG-16 [57], ResNet-101 [27] and Xception [14]. In this work, we implement DeepLab-v3+ using Xception as the base network. The Xception architecture enjoys the simplicity of VGG with multiple convolution layers stacked on top of one another. Furthermore, Xception modules use skip connections similar to ResNet and are also based on the Inception [61] hypothesis which postulates the separation of convolution operations performed on spatial (height and width) dimensions and those on the depthwise (or cross-channel) dimensions.

3.2. Approximate inference in DeepLab-v3+

In Section 2.2 we listed some techniques for approximate inference in deep neural networks. The purpose of this work is to develop and present metrics with which these inference techniques can be evaluated and benchmarked for the task of semantic segmentation. To do this, we have to first create a probabilistic deep neural network for semantic segmentation. As mentioned before, we use DeepLab-v3+ as the network of choice. Furthermore, we use MC dropout [21] as the primary method of approximate inference. It is one of the current standard baselines and has already been applied to multiple problems in computer vision [32, 34, 33].

MC dropout works on the principle of variational inference. The idea is to get a posterior distribution $p(\mathbf{W}|\mathbf{X}, \mathbf{Y})$ over the weights $\mathbf{W}$ of the neural network, given the training samples $\mathbf{X}$ and the corresponding labels $\mathbf{Y}$. However, the posterior is intractable and hence, an approximation to the posterior $q(\mathrm{W})$ known as the variational distribution is defined and the Kullback-Leibler (KL) divergence:

$$\mathrm{KL}(q(\mathrm{W})||p(\mathbf{W}|\mathbf{X}, \mathbf{Y})) \tag{1}$$

between the actual posterior and the variational distribution is minimized. In MC dropout, the variational distribution defined over the network weights is a Bernoulli distribution. In [21], the authors observed that placing a Bernoulli distribution with parameter p_b over the weights of a hidden layer is equivalent to performing dropout on that layer with a dropout rate of p_b. Furthermore, they also noted that minimising the well-known cross-entropy loss function using standard optimisation algorithms like stochastic gradient descent has the desired effect of minimizing the KL divergence term in equation 1. Hence, in order to perform approximate inference, one first needs to train a network with dropout. However, unlike common practice, these dropout layers are kept active even during the test phase. The idea is to get samples from the posterior distribution and as the

dropout layers place a Bernoulli distribution over the network weights, performing a stochastic forward pass through a trained network can be interpreted as generating a Monte Carlo sample from the posterior distribution. Therefore, multiple forward passes on the same input generate multiple such Monte Carlo samples, the mean of which can then be used as the network prediction and the variance can be interpreted as an uncertainty estimate.

Ideally, in a Bayesian neural network, a dropout layer should be inserted after every hidden layer of the network. However, as was observed by Kendall et al. in [32] for the SegNet architecture and as we observe for DeepLab-v3+, insertion of dropout layers after every convolution layer in a large neural network regularises it to an extent that makes training prohibitively slow. Therefore, the dropout layers have to be inserted only in certain regions of the network. This gives rise to multiple probabilistic variants of the Bayesian DeepLab architecture depending on where in the network, the dropout layers are inserted. However, for the sake of simplicity, in this work, we insert dropout layers only in the middle flow of the DeepLab-v3+ network. We do this because of the hypothesis proposed in [32] which states that low level features in shallower layers of the network are mostly consistent across the distribution of models and hence can be represented using deterministic weights whereas the higher level features in deeper layers are better modelled using probabilistic weights.

With the incorporation of MC dropout in DeepLab-v3+, we end up with Bayesian DeepLab, a probabilistic deep neural network designed specifically for semantic segmentation. However, in this work, our goal is to develop evaluation metrics for such networks and lay out a few benchmarks based on these metrics. Thus, in order to compare performance with MC dropout, we implement an alternative inference technique, namely Concrete dropout [22]. We choose Concrete dropout in particular because both the approximate inference techniques are dropout based methods. Furthermore, they are very simple to implement and require minimal changes to the network architecture. Concrete Dropout is a modification on the MC dropout method where the network tunes the dropout rates as part of the optimisation process. Hence, we can have a Concrete dropout layer after every convolutional layer in the network instead of having to select certain parts of the network to insert the dropout layer in.

3.3. Network Architecture

The backbone framework of Bayesian DeepLab is similar to DeepLab-v3+ [13] where the inputs are first passed through an extended Xception [14] network followed by an ASPP module for multi-scale image processing and finally a decoder module to resize the images to the original input dimensions and to produce sharp segmentation results. The differences between the network architecture which we use in this work and the original DeepLab-v3+ network proposed in [13] are as follows:

1. Bayesian DeepLab has dropout layers at different points in the network. In particular we insert a dropout layer after every 4 Xception modules in the middle flow of the network. As there are 16 Xception modules in the middle flow, there are a total of 4 dropout layers in the network. We use a dropout rate of 0.5 in each of the dropout layers. However, these rates are hyperparameters which can be fine-tuned further.
2. We do not use cascaded atrous modules or image pyramids in our network. The only method for multi-scale image processing adopted by Bayesian DeepLab is Atrous Spatial Pyramid Pooling (ASPP). We do this primarily for the sake of simplicity and to reduce training time.

We present the Bayesian DeepLab architecture in Figure 7 in the appendix.

3.4. Uncertainty Metrics

There are two types of uncertainties which we study in this work. *Epistemic uncertainty*, also known as *model uncertainty* represents what the model does not know due to insufficient training data. This kind of uncertainty can be explained away with more training data. *Aleatoric uncertainty* is caused due to noisy measurements in the data and can be explained away with increased sensor precision (but cannot be explained away with increase in training data). The two uncertainties combined form the *predictive uncertainty* of the network. In [20], the author suggests some information theoretic metrics which can be used as measures of uncertainty in classification problems. In our work, we use two such metrics, namely the *entropy* of the predictive distribution (also known as *predictive entropy*) and the *mutual information* between the predictive distribution and the posterior over network weights.

The predictive entropy $\hat{\mathbb{H}}[y|\mathbf{x}, \mathcal{D}_{train}]$ given a test input $\mathbf{x}$ and the training data $\mathcal{D}_{train}$ can be approximated as follows:

$$\hat{\mathbb{H}}[y|\mathbf{x}, \mathcal{D}_{train}] = \quad (2)$$
$$-\sum_{c}\left(\frac{1}{T}\sum_{t} p(y=c|\mathbf{x}, \hat{w}_t)\right)\log\left(\frac{1}{T}\sum_{t} p(y=c|\mathbf{x}, \hat{w}_t)\right)$$

where y is the output variable, c ranges over all the classes, T is the number of Monte Carlo samples (stochastic forward passes), $p(y = c|\mathbf{x}, \hat{w}_t)$ is the softmax probability of input $\mathbf{x}$ being in class c, and $\hat{w}_t$ are the model parameters on the t^{th} Monte Carlo sample. Similarly, the mutual information between the predictive distribution and the posterior over

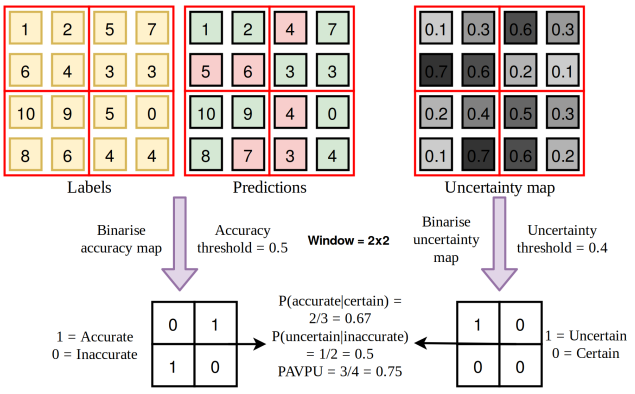


Figure 2. Worked out example of computing the performance evaluation metrics for Bayesian models in semantic segmentation.

model parameters can be approximated as:

$$\hat{\mathbb{I}}[y, w|\mathbf{x}, \mathcal{D}_{train}] = \hat{\mathbb{H}}[y|\mathbf{x}, \mathcal{D}_{train}] + \frac{1}{T}\sum_{c,t} p(y = c|\mathbf{x}, \hat{w}_t) \log p(y = c|\mathbf{x}, \hat{w}_t). \quad (3)$$

We choose the predictive entropy and mutual information metrics as they capture different kinds of uncertainty. As observed in [20], mutual information captures epistemic or model uncertainty whereas predictive entropy captures predictive uncertainty which combines both epistemic and aleatoric uncertainties. It is worth noting here that in semantic segmentation, we produce pixel-wise classification results and hence, we also produce pixel-wise uncertainty estimates. Thus, we end up with segmentation results and uncertainty maps which have the same dimensions as that of the input image.

4. Performance Evaluation Metrics

As mentioned in Section 1, we measure the performance of Bayesian models using metrics which capture properties which we want the model to satisfy. In particular, we assume that *if a model is confident about its prediction, it should be accurate on the same.* This also implies that *if a model is inaccurate on an output, it should be uncertain about the same output.* It is worth noting that the converse of these assumptions may not hold. For instance, a model may have a high epistemic uncertainty on a class which appears infrequently in the training set but can still be accurate on its prediction. Given the above assumptions, we can define the following two conditional probabilities:

1. $p(\mathbf{accurate}|\mathbf{certain})$: The probability that the model is accurate on its output given that it is confident on the same.

2. $p(\mathbf{uncertain}|\mathbf{inaccurate})$: The probability that the model is uncertain about its output given that it has made a mistake in its prediction (i.e., is inaccurate).

In order to implement the above metrics, we first choose a patch/window size w and traverse the predicted labels, actual labels and uncertainty maps using windows of dimensions $w \times w$, much like the traversal in a convolution operation. Technically, the patch dimensions can be as small as a single pixel (i.e., a 1×1 patch) or as large as the entire image. However, labelling the entire image as accurate or uncertain is not very useful. Furthermore, the uncertainties occur in regions within the image comprising multiple pixels in close neighbourhoods. Hence, having 1×1 single pixel patches is also not useful. We compute the accuracy of each such patch from the predicted and actual labels. This accuracy can be computed using any standard technique. In this work we use the *pixel accuracy* metric defined in [46]. If the patch accuracy is above a certain threshold, we mark the patch as *accurate*.

Similarly, from the corresponding patch obtained from the uncertainty map, we compute the average patch uncertainty. If this uncertainty value is above a given threshold, we label the patch as *uncertain*. There can be multiple ways of setting the uncertainty threshold. One simple way would be to find the average uncertainty of all pixels over a validation set and use that value as the threshold. Other ways could include computing the minimum u_{min} and maximum u_{max} uncertainty values over validation set pixels and setting the uncertainty threshold u_{th} as:

$$u_{th} = u_{min} + (t(u_{max} - u_{min})). \quad (4)$$

where t is in $[0, 1]$. Once the entire dimensions of the image have been covered, we construct a confusion matrix containing the number of patches which are accurate and certain (n_{ac}), accurate and uncertain (n_{au}), inaccurate and certain (n_{ic}) and inaccurate and uncertain (n_{iu}). We can then report the conditional probabilities $p(\mathbf{accurate}|\mathbf{certain})$ and $p(\mathbf{uncertain}|\mathbf{inaccurate})$ as follows:

$$p(\mathbf{accurate}|\mathbf{certain}) = \frac{n_{ac}}{(n_{ac} + n_{ic})} \quad (5)$$

$$p(\mathbf{uncertain}|\mathbf{inaccurate}) = \frac{n_{iu}}{(n_{ic} + n_{iu})} \quad (6)$$

Finally, we combine both the good cases of (accurate, certain) and (inaccurate, uncertain) patches into a single metric, the *Patch Accuracy vs Patch Uncertainty* (PAvPU), defined as follows:

$$\text{PAvPU} = \frac{(n_{ac} + n_{iu})}{(n_{ac} + n_{au} + n_{ic} + n_{iu})} \quad (7)$$

Clearly, a model with a higher value of the above metrics is a better performer. It should be noted that the values of

Guaranteed to be n

the above metrics are dependent on three parameters: the accuracy threshold, the uncertainty threshold, and the patch dimensions. An interesting experiment would be to observe how the values of the metrics change with these parameters. We show this in Figures 3 and 4. In Figure 2, we provide an illustrative toy example of computing the above three metrics.

5. Experiments and Results

In this section we first evaluate the trained models on their segmentation performance using the pixel accuracy, mean accuracy and mean IOU metrics defined in [46]. Next, we compare and benchmark MC dropout [21] and Concrete dropout [22] inference using the metrics proposed in Section 4. All experiments have been performed on Cityscapes [15], one of the most popular datasets for urban scene understanding. It has 5000 images collected from street scenes in 50 different cities. There are 2975 images in the training set, 500 images in the validation set and 1525 test images with dimensions 2048×1024. We provide details about the training infrastructure in Appendix B.

5.1. Semantic Segmentation Performance

In Table 1 we report the semantic segmentation results obtained from the Bayesian DeepLab variants using both MC dropout and Concrete dropout inference and compare these results with different versions of DeepLab [10, 11, 13] on the Cityscapes *val* set. We observe that MC dropout performs better than Concrete dropout with respect to all the three metrics: pixel accuracy, mean accuracy and mean IOU. Furthermore, both our models with mean IOU values of 78.05 and 77.53 outperform all the DeepLab versions except DeepLab-v3+ which has a mean IOU of 79.14. It is worth noting that we compare our models only with those versions of DeepLab which use the same hyperparameters as us. To be precise, we compare with the version of DeepLab-v3 which uses an output stride of 16 and DeepLab-v3+ which uses the Xception-65 network with ASPP and decoder modules but without image level features.

In Figure 5, we present some qualitative results on Cityscapes *val* set images. It is interesting to note the difference between the uncertainty maps provided by the predictive entropy and mutual information metrics. In case of mutual information, we observe high uncertainty inside the boundaries of objects which the model is confused about. The bonnet of the Mercedes at the bottom of each image is one such example. However, in predictive entropy maps, we also see high uncertainties on the edges of objects like pedestrians or cars. These edges are regions where the presence of noise in the dataset is highly likely. This observation supports the explanation in [20] that mutual information captures epistemic or model uncertainty and predictive entropy captures aleatoric uncertainty.

Method	Pixel Accuracy	Mean Accuracy	Mean IOU
DeepLab			
DeepLab (VGG-16) [10]	NA	NA	65.94
DeepLab (ResNet-101) [10]	NA	NA	71.40
DeepLab-v3 (OS=16) [11]	NA	NA	77.23
DeepLab-v3+ (X-65) [13]	NA	NA	79.14
Bayesian DeepLab			
MC Dropout	95.31	85.11	78.05
Concrete Dropout	94.81	83.24	77.53

Table 1. **Semantic segmentation performance on Cityscapes *val* set for Bayesian DeepLab networks.** The performance is measured using three well-known metrics: pixel accuracy, mean accuracy and mean IOU [46].

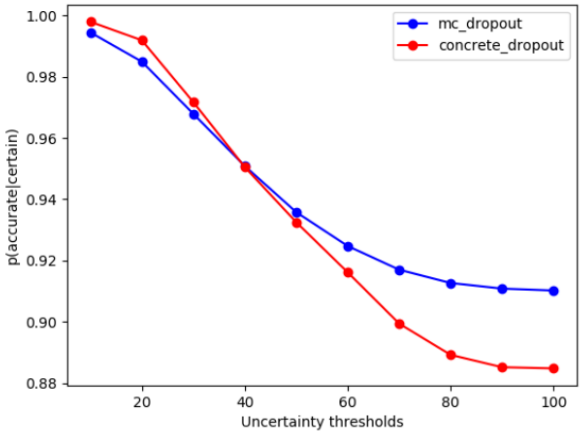
5.2. Evaluation of Bayesian DeepLab

In order to compute the metrics $p(\mathbf{accurate}|\mathbf{certain})$, $p(\mathbf{uncertain}|\mathbf{inaccurate})$ and PAvPU, we need to set three parameters: the uncertainty and accuracy thresholds and the patch dimensions. In our experiments, we use patches with dimensions 4×4 although other patch dimensions can also be used. For the sake of simplicity, we set the accuracy threshold to 0.5 (or 50%) and the uncertainty threshold to the mean uncertainty value in the validation set.

As seen in Table 2, MC dropout consistently outperforms Concrete dropout on all the metrics. In Figures 3 and 4, we plot these metrics for varying thresholds of uncertainty. We can draw the following observations from the plots:

1. When the uncertainty threshold is 0, all the patches are marked uncertain. Hence $n_{ac}+n_{ic} = 0$ in equation 5 and the value of $p(\mathbf{accurate}|\mathbf{certain})$ is undefined. For the plot of $p(\mathbf{accurate}|\mathbf{certain})$ we start the uncertainty threshold from 10%. Furthermore, $n_{ic} + n_{iu} = n_{iu}$ in equation 6 and therefore, $p(\mathbf{uncertain}|\mathbf{inaccurate})$ is 1. Lastly, as $n_{ac} = n_{ic} = 0$, the PAvPU metric (as defined in equation 7) becomes $\frac{n_{iu}}{n_{iu}+n_{au}}$ which in this case, is the fraction of inaccurate patches in the image.
2. When the uncertainty threshold is 100%, all the patches are labelled certain. In this case, both $p(\mathbf{accurate}|\mathbf{certain})$ and PAvPU boil down to the fraction of patches which are accurate. As $n_{iu} = 0$, $p(\mathbf{uncertain}|\mathbf{inaccurate})$ reduces to 0.
3. It is interesting to note that the sum of the PAvPU values at uncertainty thresholds 0 and 100% is 1. Furthermore,

AND epistemic u



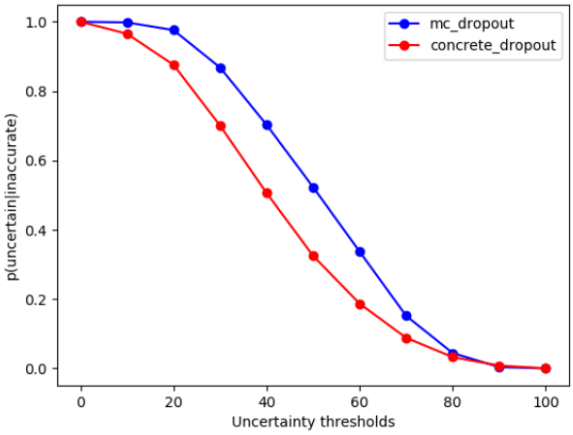


Figure 3. **Plots of p(accurate|certain) and p(uncertain|inaccurate) for varying thresholds of uncertainty.**

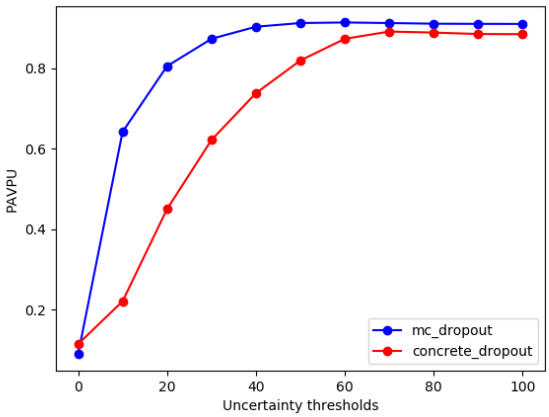


Figure 4. **Plot of PAvPU for varying thresholds of uncertainty.**

Method	p(accurate| certain)	p(uncertain| inaccurate)	PAvPU
Predictive Entropy			
MC Dropout	0.9869	0.8962	0.7861
Concrete Dropout	0.9602	0.7861	0.6854
Mutual Information			
MC Dropout	0.9630	0.6720	0.8267
Concrete Dropout	0.9266	0.6241	0.7553

Table 2. Performance of Bayesian DeepLab variants evaluated using the three metrics proposed in Section 4.

the PAvPU value at 100% uncertainty threshold is significantly greater than the value at 0. This indicates that the number of accurate patches is much higher than the number of inaccurate patches.

4. Finally, as a higher value for the metrics indicates better performance, all the three plots indicate the superior performance of MC dropout over Concrete dropout which is consistent with the results in Table 2.

In Figure 6 we show the computation of the the $p(\mathbf{accurate}|\mathbf{certain})$, $p(\mathbf{uncertain}|\mathbf{inaccurate})$ and PAvPU metrics for real world images using predictive entropy. We have two successful cases where the network makes accurate predictions and two failure cases where the network misclassifies a truck as a bus. We can see from the predictive entropy that the model is uncertain on the misclassified pixels. For every case, we have two binary maps, one for accuracy and one for uncertainty. Each pixel in a binary map corresponds to a patch in the image. For the accuracy/uncertainty maps, a white pixel represents an accurate/uncertain patch and a black pixel represents the opposite. It is interesting to note how these maps show the inverse relation between model accuracy and uncertainty. However, in the two failure cases, we also observe that n_{au} and n_{ic} values (representing the undesirable patches) are relatively high. **This gets reflected in the values of the metrics which are lower for the failure cases compared to the successful ones**.

6. Conclusions

In this work we have developed metrics to evaluate Bayesian models for the task of semantic segmentation. We have created two probabilistic variants of the DeepLab-v3+ [13] network and have evaluated them using these metrics, thereby providing benchmarks which can be used for future comparisons. For experiments, we have used the Cityscapes

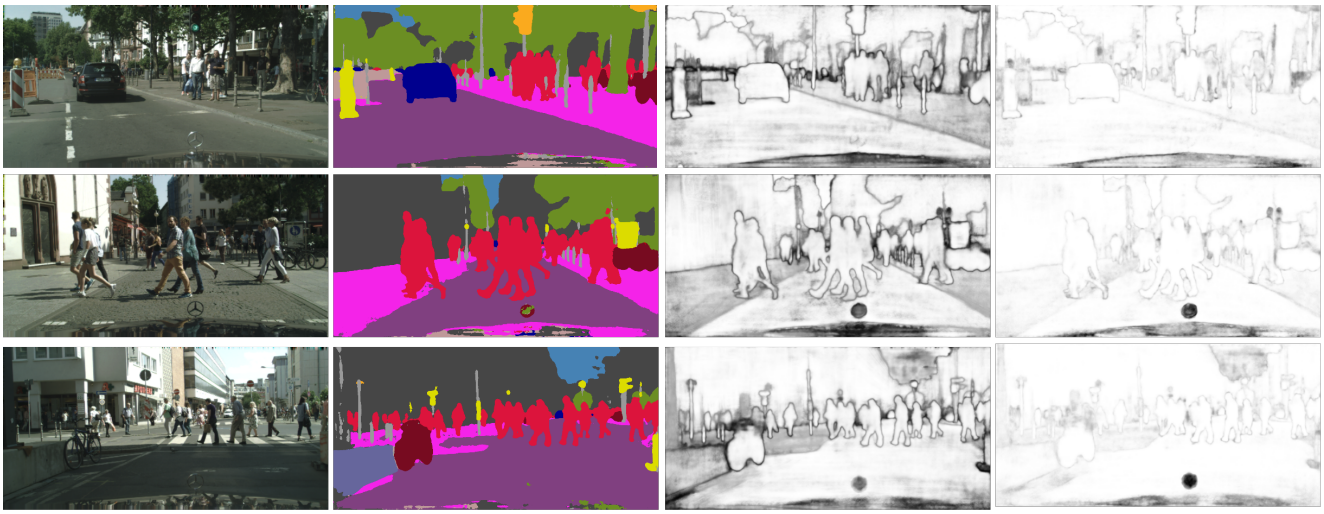


Figure 5. **Qualitative results for semantic segmentation with uncertainty estimates on Cityscapes images.** The results include images from the Cityscapes *val* set, the corresponding semantic segmentation results from our model and the predictive and epistemic uncertainties estimated through the predictive entropy and the mutual information metrics respectively. Darker shades indicate higher uncertainty.

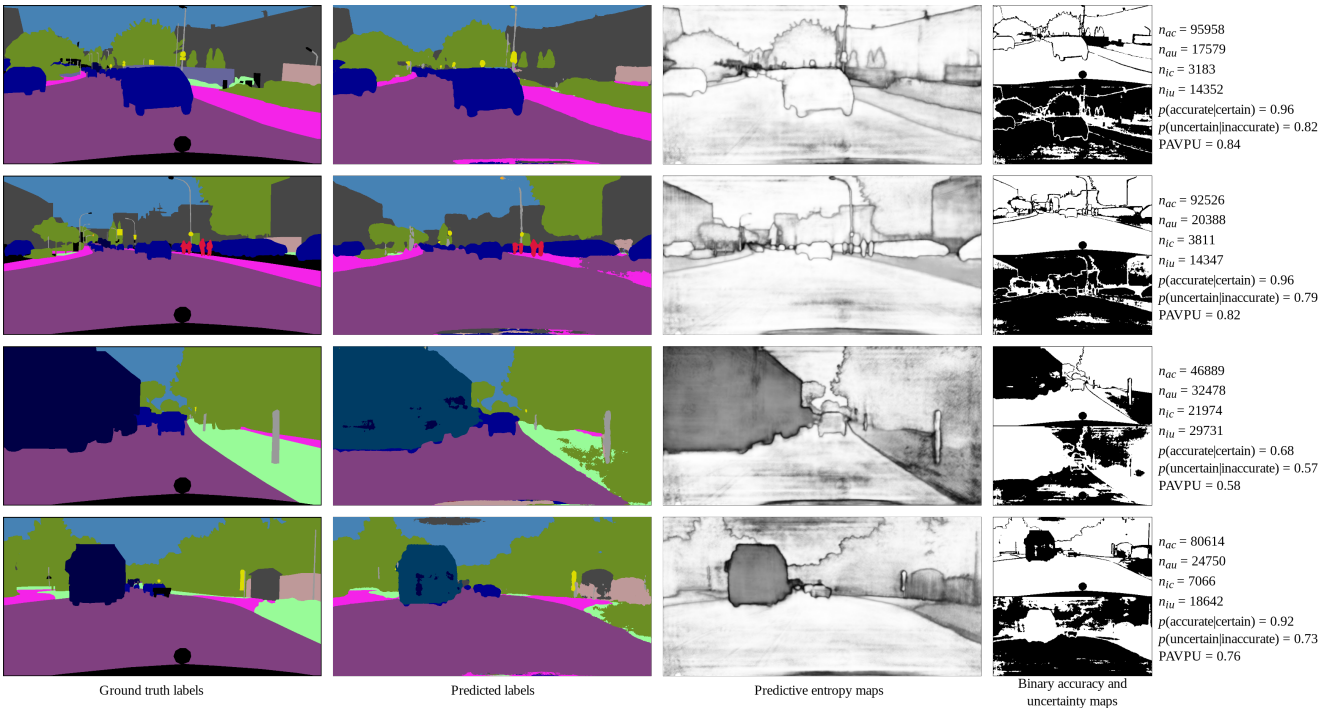


Figure 6. **Computation of $p(\mathbf{accurate}|\mathbf{certain})$, $p(\mathbf{uncertain}|\mathbf{inaccurate})$ and PAvPU on real world images.** We take two success and two failure cases and compute the metrics for each of these cases. The upper binary map in each row is the accuracy map and the lower map corresponds to the uncertainty map. White regions in these accuracy/uncertainty maps represent accurate/uncertain patches.

[15] dataset which is particularly suited for applications like autonomous driving. An interesting future work would be to develop metrics which measure performance of Bayesian models based on how effective the segmentation outputs and uncertainty estimates are in making safe and correct autonomous driving decisions. This idea can be further extended to include other downstream applications as well where semantic segmentation is a useful intermediate tool.

References

[1] S. Ahn, A. Korattikara, and M. Welling. Bayesian posterior sampling via stochastic gradient fisher scoring. *arXiv preprint arXiv:1206.6380*, 2012. 3

[2] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 33(8):831, 2015. 1

[3] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning*, pages 173–182, 2016. 1

[4] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016. 1

[5] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (12):2481–2495, 2017. 1, 2, 3

[6] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015. 1, 3

[7] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016. 1

[8] T. Bui, D. Hernández-Lobato, J. Hernandez-Lobato, Y. Li, and R. Turner. Deep gaussian processes for regression using approximate expectation propagation. In *International Conference on Machine Learning*, pages 1472–1481, 2016. 1

[9] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014. 1, 2, 3

[10] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2018. 1, 2, 3, 6

[11] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 1, 2, 3, 6, 13

[12] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille. Attention to scale: Scale-aware semantic image segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3640–3649, 2016. 3

[13] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *arXiv preprint arXiv:1802.02611*, 2018. 1, 2, 3, 4, 6, 7, 13

[14] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 3, 4, 12

[15] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 1, 6, 7

[16] G. E. Dahl, D. Yu, L. Deng, and A. Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, 20(1):30–42, 2012. 1

[17] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115, 2017. 1

[18] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2013. 1

[19] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2013. 3

[20] Y. Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016. 4, 5, 6

[21] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR. 1, 2, 3, 6

[22] Y. Gal, J. Hron, and A. Kendall. Concrete Dropout. In *Advances in Neural Information Processing Systems 30 (NIPS)*, 2017. 1, 2, 3, 4, 6, 13

[23] A. Giusti, D. C. Ciresan, J. Masci, L. M. Gambardella, and J. Schmidhuber. Fast image scanning with deep max-pooling convolutional neural networks. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 4034–4038. IEEE, 2013. 3

[24] A. Graves. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356, 2011. 3

[25] A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013. 1

[26] A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649. IEEE, 2013. 1

[27] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[28] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012. 1

[29] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In J.-M. Combes,

A. Grossmann, and P. Tchamitchian, editors, *Wavelets*, pages 286–297, Berlin, Heidelberg, 1990. Springer Berlin Heidelberg. 3

[30] B. Huval, T. Wang, S. Tandon, J. Kiske, W. Song, J. Pazhayampallil, M. Andriluka, P. Rajpurkar, T. Migimatsu, R. Cheng-Yue, et al. An empirical evaluation of deep learning on highway driving. *arXiv preprint arXiv:1504.01716*, 2015. 1

[31] K. Kavukcuoglu, P. Sermanet, Y.-L. Boureau, K. Gregor, M. Mathieu, and Y. L. Cun. Learning convolutional feature hierarchies for visual recognition. In *Advances in neural information processing systems*, pages 1090–1098, 2010. 1

[32] A. Kendall, V. Badrinarayanan, , and R. Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*, 2015. 2, 3, 4

[33] A. Kendall and Y. Gal. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In *Advances in Neural Information Processing Systems 30 (NIPS)*, 2017. 3

[34] A. Kendall, Y. Gal, and R. Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[35] D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, pages 2575–2583, 2015. 3

[36] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1

[37] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017. 1

[38] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196, 2014. 1

[39] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 336–345, 2017. 1

[40] M. K. Leung, H. Y. Xiong, L. J. Lee, and B. J. Frey. Deep learning of the tissue-regulated splicing code. *Bioinformatics*, 30(12):i121–i129, 2014. 1

[41] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 163–168. IEEE, 2011. 1

[42] G. Lin, A. Milan, C. Shen, and I. Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2

[43] G. Lin, C. Shen, A. Van Den Hengel, and I. Reid. Efficient piecewise training of deep structured models for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3194–3203, 2016. 3

[44] S. Liu, H. Zheng, Y. Feng, and W. Li. Prostate cancer diagnosis using deep learning with 3d multiparametric mri. In *Medical Imaging 2017: Computer-Aided Diagnosis*, volume 10134, page 1013428. International Society for Optics and Photonics, 2017. 1

[45] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang. Semantic image segmentation via deep parsing network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1377–1385, 2015. 3

[46] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 1, 2, 5, 6, 12

[47] C. Louizos and M. Welling. Structured and efficient variational deep learning with matrix gaussian posteriors. In *International Conference on Machine Learning*, pages 1708–1716, 2016. 1

[48] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. 1

[49] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013. 1

[50] A.-r. Mohamed, G. E. Dahl, G. Hinton, et al. Acoustic modeling using deep belief networks. *IEEE Trans. Audio, Speech & Language Processing*, 20(1):14–22, 2012. 1

[51] R. M. Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012. 3

[52] G. Papandreou, I. Kokkinos, and P.-A. Savalle. Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 390–399, 2015. 3

[53] P. H. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene labeling. In *31st International Conference on Machine Learning (ICML)*, number EPFL-CONF-199822, 2014. 3

[54] M. I. Razzak, S. Naz, and A. Zaib. Deep learning for medical image processing: Overview, challenges and the future. In *Classification in BioApps*, pages 323–350. Springer, 2018. 1

[55] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 1, 2, 3

[56] S. Schulter, P. Vernaza, W. Choi, and M. Chandraker. Deep network flow for multi-object tracking. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 2730–2739. IEEE, 2017. 1

[57] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

[58] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013. 1

[59] K. Suzuki. Overview of deep learning in medical imaging. *Radiological physics and technology*, 10(3):257–273, 2017. 1

[60] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 1

[61] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 3

[62] Y. Tang. Deep learning using linear support vector machines. *arXiv preprint arXiv:1306.0239*, 2013. 1

[63] M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688, 2011. 3

[64] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015. 2

[65] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015. 3

[66] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890, 2017. 1, 3

Appendices

A. Bayesian DeepLab Network Architecture

In Figure 7, we present the network architecture of Bayesian DeepLab using MC dropout inference. We use Xception as the base network. However, there are certain differences between the original Xception [14] architecture and the one which we use. The differences are as follows:

1. There are no pooling layers in our network. We use separable convolution filters with a stride of 2 instead of max pooling layers. This helps in dense predictions. Furthermore, following the FCN [46] philosophy, our network is fully convolutional and hence can segment images of arbitrary sizes.

2. The middle flow in our network has 16 modules instead of 8 as described in the original Xception paper [14].

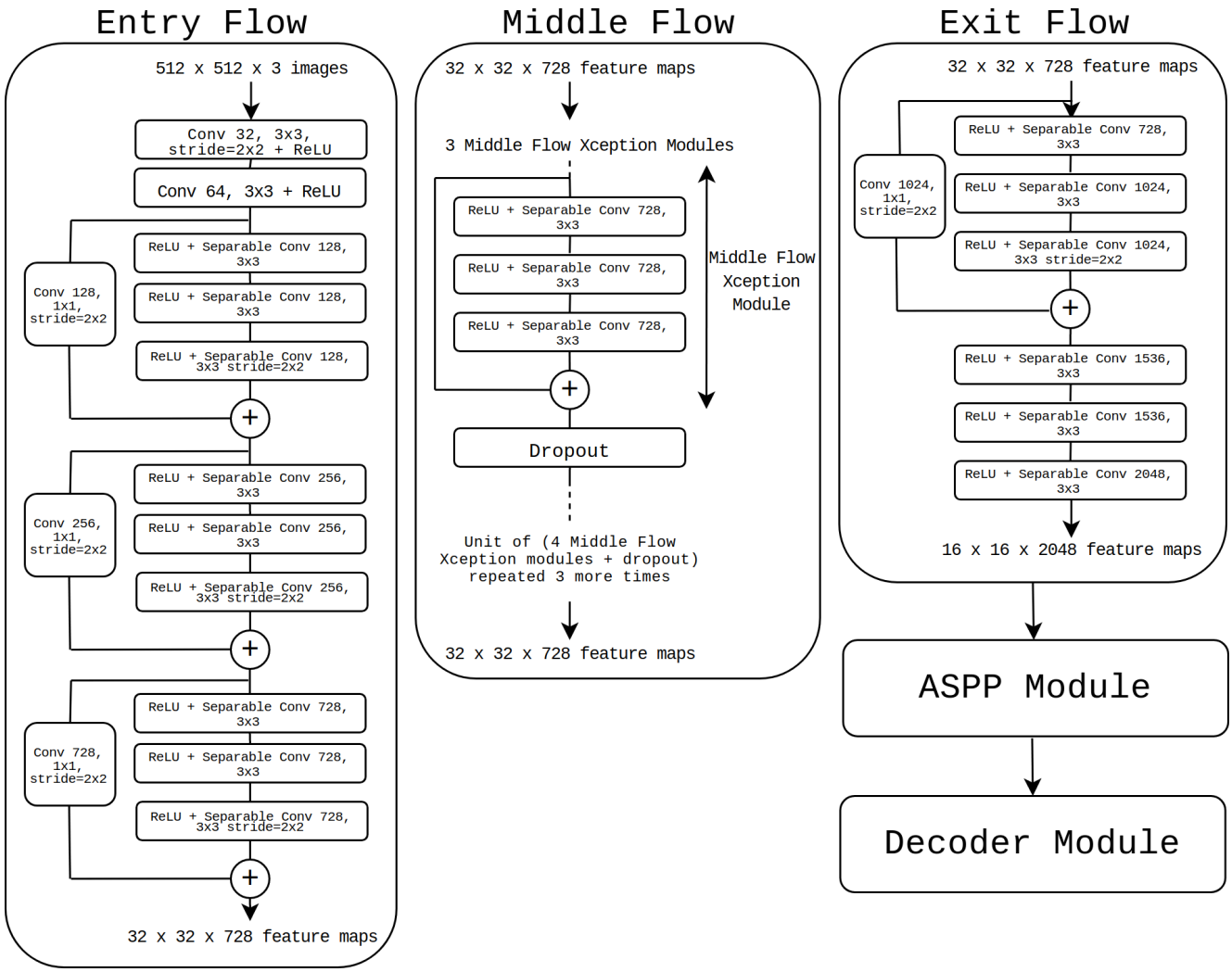


Figure 7. **Bayesian DeepLab network architecture:** The inputs are passed in order first through the entry flow followed by the middle flow and finally through the exit flow. The ASPP module is used to recognize objects at different scales and the Decoder module resizes the outputs to the original input dimensions.

B. Training Infrastructure

In order to train the Bayesian DeepLab network, we set the following parameters:

1. a list of *atrous rates* for the ASPP module which we set to $[6, 12, 18]$ following the DeepLab-v3+ [13] paper,
2. the *output stride* and the *decoder output stride* [11] which we set as 16 and 4 respectively,
3. the *crop size* for training images which we set to 512×512 and
4. the *training batch size* which we set to 16.

Furthermore, the Concrete dropout layers require two additional hyperparameters: the *weight regulariser* which is set to $1e-8$ and the *dropout regulariser* which we set to $1/(n \times h \times w)$, where n is the number of training images and h and w are the height and width of the image respectively, following the paper [22]. We train all the networks for 90,000 iterations on 8 NVIDIA Tesla P100-SXM2 GPUs. The networks finish training in approximately 3 days.