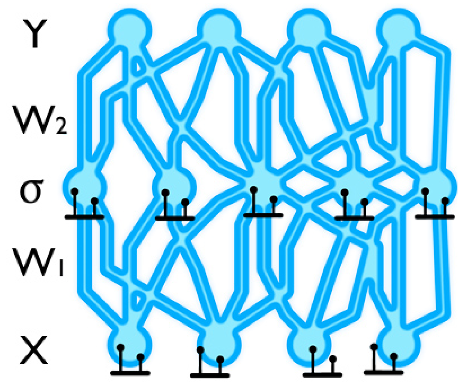
# Concrete Dropout

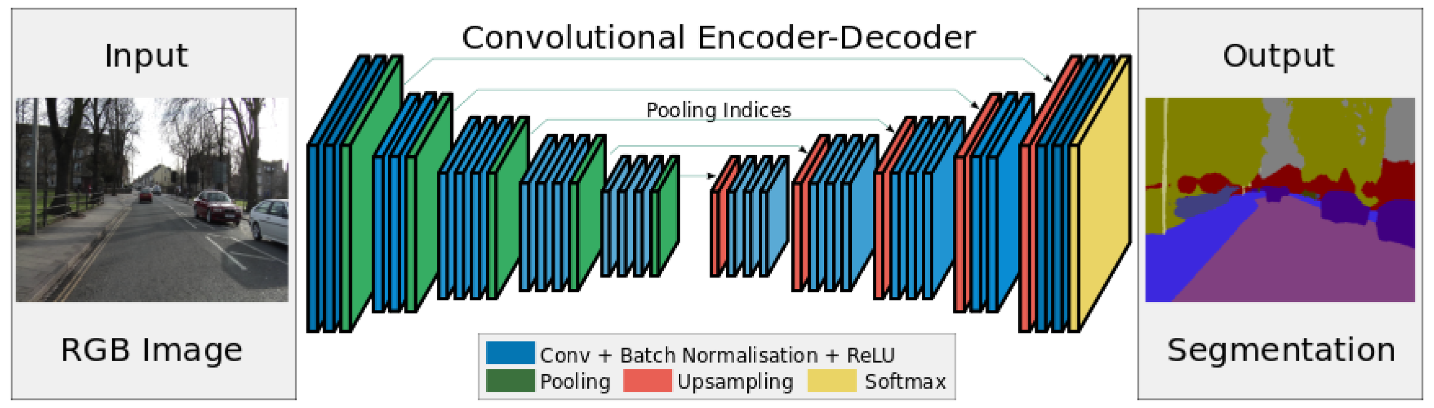
Yarin Gal[1,2,3], Jiri Hron[1], Alex Kendall[1]

1: Department of Engineering, University of Cambridge, UK 2: Alan Turing Institute, UK, 3: Department of Computer Science, University of Oxford, UK

## Motivation

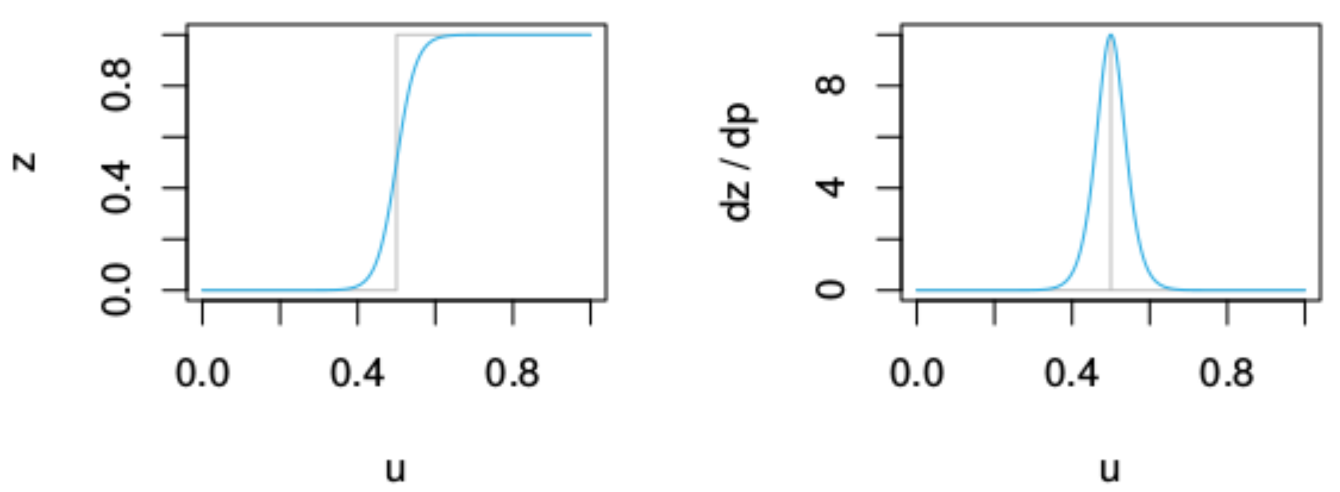
- **Dropout probabilities** have significant effect on predictive performance
- Traditional grid search or manual **tuning is prohibitively expensive** for large models
- **Optimisation** wrt a sensible objective should result in better **calibrated uncertainty**, and **shorter experiment cycle**
- Useful for large modern models in machine vision and reinforcement learning




## Background

- Gal and Gharamani (2015) reinterpreted dropout regularisation as approximate inference in BNNs
- Dropout probabilities $p_l$ are variational parameters of the approximate posterior $q_\theta(\boldsymbol{\omega}) = \prod_k q_{\boldsymbol{M}_k,p_k}(\boldsymbol{W}_k)$, where $\boldsymbol{W}_k = \boldsymbol{M}_k \cdot \mathrm{diag}(\boldsymbol{z}_k)$ and $\boldsymbol{z}_{kl} \overset{\text{iid}}{\sim} \mathrm{Bernoulli}(1-p_k)$
- Concrete distribution (Maddison et al., Jang et al.) relaxes Categorical distribution to obtain gradients wrt the probability vector
  - Example: $\boldsymbol{z}_{lk} \overset{\text{iid}}{\sim} \mathrm{Bernoulli}(1-p_k)$ is replaced by $\tilde{\boldsymbol{z}}_{kl} = \mathrm{sigmoid}\left((\log \frac{p_k}{1-p_k} + \log \frac{u_{kl}}{1-u_{kl}})/t\right)$ where $u_{kl} \overset{\text{iid}}{\sim} \mathrm{Uniform}(0,1)$




## Learning dropout probabilities

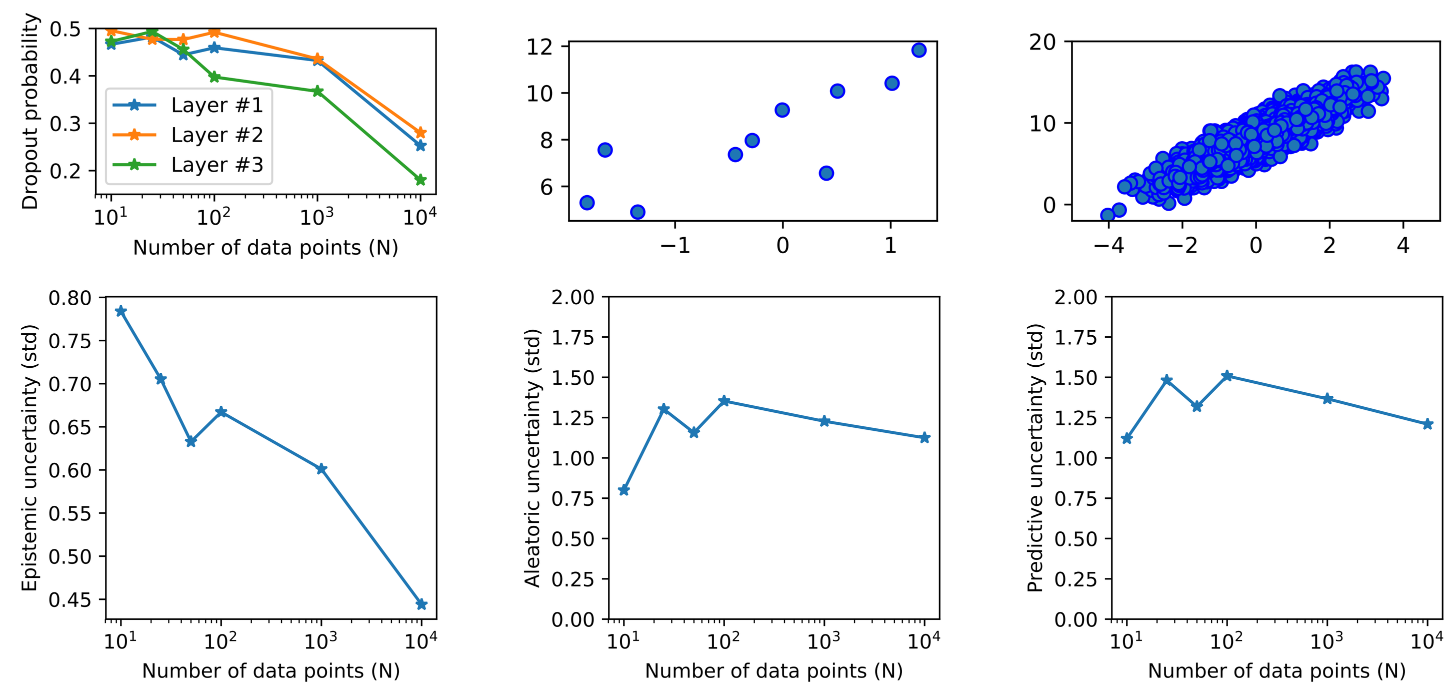
SVI (Hoffman et al., 2013) can be used to approximate the posterior:

$$\widehat{\mathcal{L}}_{\text{MC}}(\theta) = -\frac{1}{M}\sum_{i\in S}\log p(\boldsymbol{y}_i \mid \boldsymbol{f}_{\omega_\theta}(\boldsymbol{x}_i)) + \frac{1}{N}\mathrm{KL}\left(q_\theta(\boldsymbol{\omega}) \| p(\boldsymbol{\omega})\right)$$

Structure of $q_\theta(\boldsymbol{\omega})$ turns calculation of the KL into a sum over:

$$\mathrm{KL}\left(q_{\boldsymbol{M}_k,p_k}(\boldsymbol{W}_k) \| p(\boldsymbol{W}_k)\right) \propto \frac{l^2(1-p_k)}{2}\|\boldsymbol{M}_k\|_F^2 - K_{k+1}\mathcal{H}(p_k)$$

$$\mathcal{H}(p_k) := -p_k \log p_k - (1-p_k)\log(1-p_k)$$




Properties:

- For large $K_l$, $\mathcal{H}(p_k)$ pushes $p_k \to 0.5$, maximising entropy
- Large $\|\boldsymbol{M}_k\|_F^2$ forces $p_k$ to 1, i.e. to drop all weights
- As $N \to \infty$, KL is ignored and posterior concentrates at MLE

Simple to implement in Keras (Chollet et al., 2015):

```
# regularisation
...
kernel_regularizer = self.weight_regularizer * K.sum(K.square(weight))
dropout_regularizer = self.p * K.log(self.p) + (1.-self.p) * K.log(1.-self.p)
dropout_regularizer *= self.dropout_regularizer * input_dim
regularizer = K.sum(kernel_regularizer + dropout_regularizer)
self.add_loss(regularizer)
...
# forward pass
...
u = K.random_uniform(shape=K.shape(x))
z = K.log(self.p / (1. - self.p)) + K.log(u / (1-u))
z = K.sigmoid(z / temp)
x *= 1. - z
...
```

## Application to image segmentation

Epistemic and aleatoric uncertainty in machine vision:

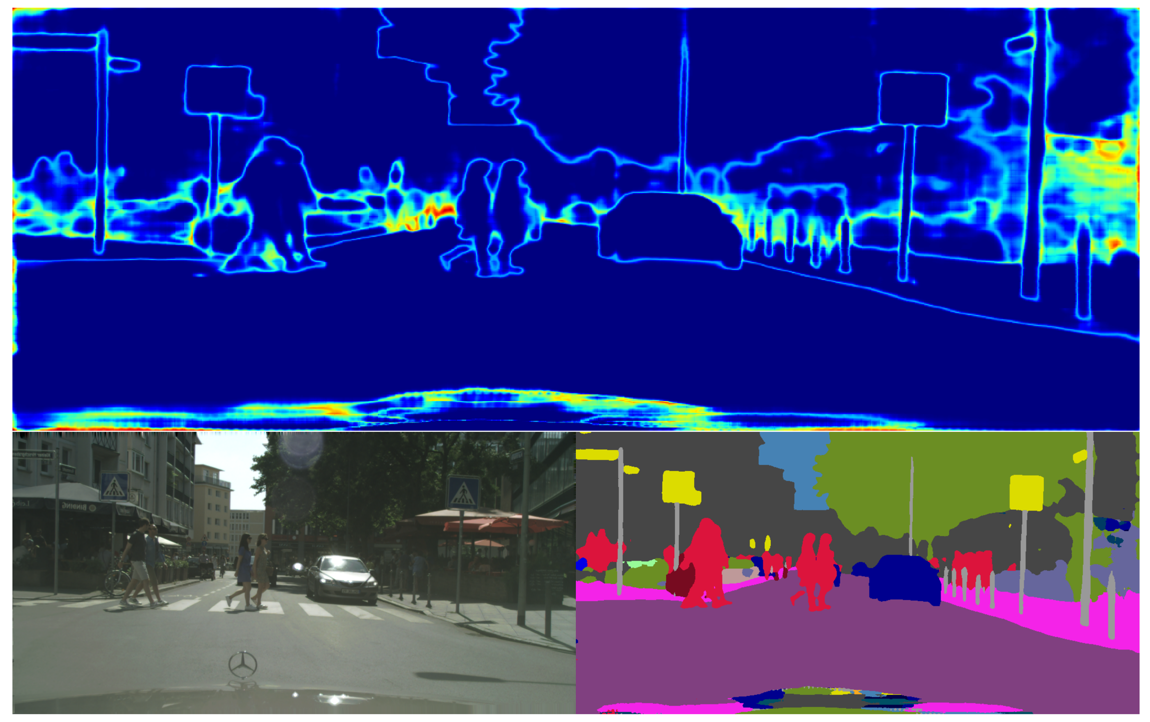
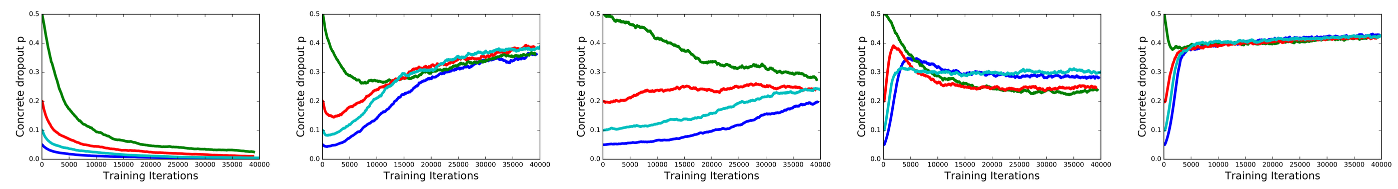


Image segmentation using Bayesian SegNet (Kendall et al., 2015)

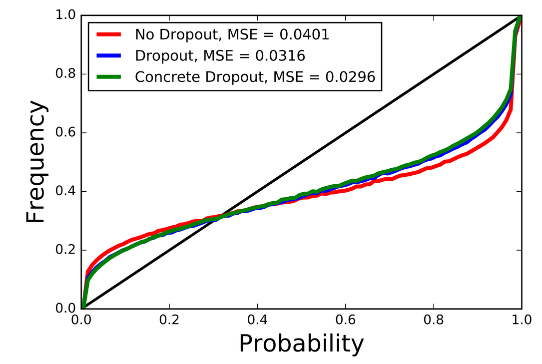
Converged probabilities were robust to random initialisation:



And compare favourably to expensively hand-tuned setting:

| DenseNet Model Variant | MC Sampling | IoU |
|---|---|---|
| No Dropout | - | 65.8 |
| Dropout (manually-tuned $p = 0.2$) | ✗ | 67.1 |
| Dropout (manually-tuned $p = 0.2$) | ✓ | 67.2 |
| Concrete Dropout | ✗ | 67.2 |
| Concrete Dropout | ✓ | **67.4** |

Comparing the performance against baseline models with DenseNet on the CamVid road scene semantic segmentation dataset




Reduced uncertainty calibration RMSE

## Conclusion and future research

- Tuning of dropout probabilities even for very large models
- Better calibrated uncertainty estimates
- RL: epistemic uncertainty will vanish as more data acquired