Bayesian Reparametrisation of Neural Networks

by Moritz Grünbauer

Structure

1)Motivation

2)What is Uncertainty?

3)Bayesian Neural Network

4)Original Work by Bruss, et al.

5)Reproduction

6)Results

7)Conclusion

Motivation

- Neural Networks (NN) are inherently uncertain in their results
- But they pretend to “have an answer for everything”
- If we’re able to quantify uncertainty, we can assign value to predictions of Nns
- Bayesian NNs are a way to do that, but can we make it easier?

What is Uncertainty?

Epistemic:

- Can be “trained away”
- Better Technologies and Methods can reduce this

Aleatoric:

- Inherent to the data

Example:

- Classifying 1 as 7 or vice versa
  - Because they share features

Bayesian Neural Networks

- Probability distributions rather than point estimates
- Accepting only predictions with low uncertainty

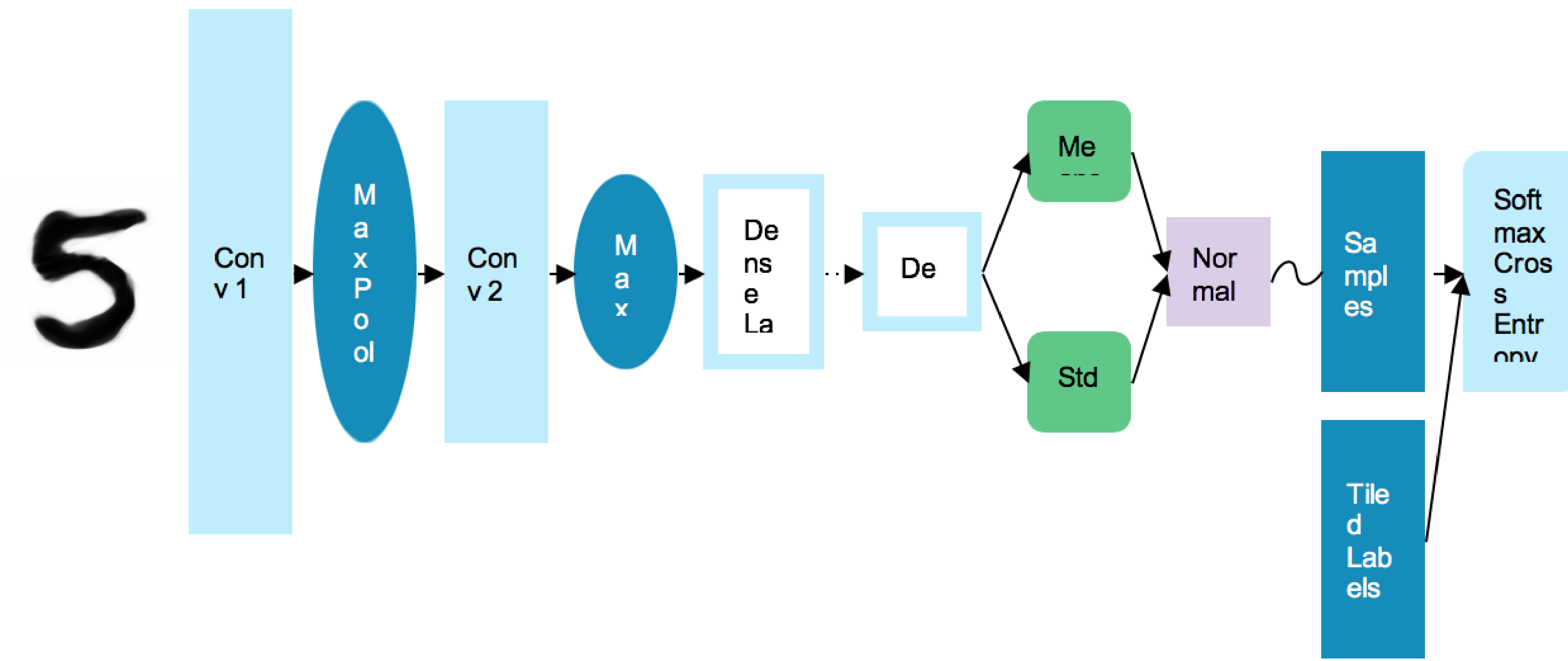
  But:

- Have to carry distributions through all layers

What if:
We could make regular NNs "Bayesian"?

Original work by Bruss, et al.

- Added reparametrisation to regular MNIST classifier to get awareness of uncertainty



Original work by Bruss, et al.

- They report: jump from 97% to 99.3% accuracy
- No mention of accepting only results with a minimum certainty

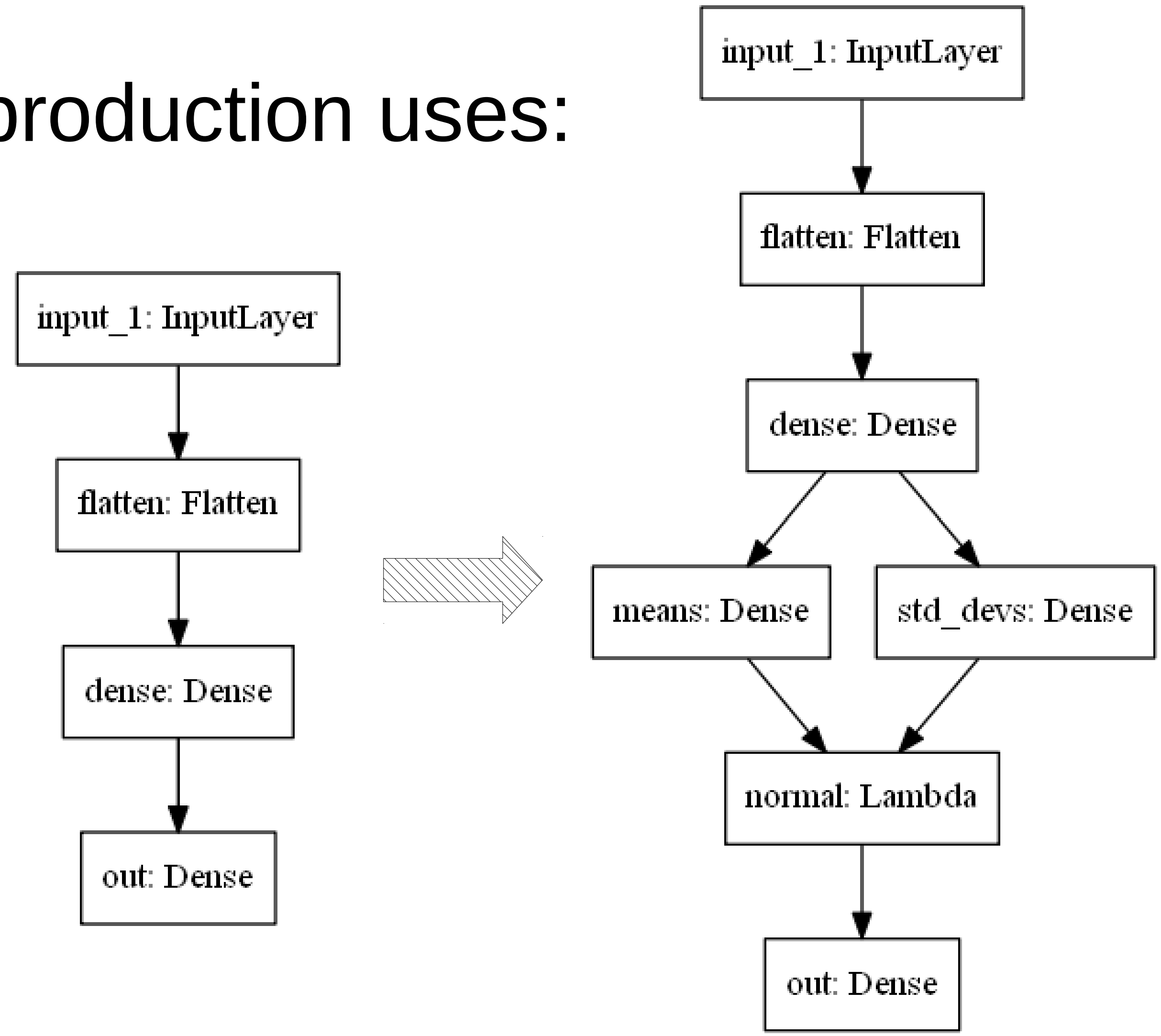
Reproduction

- No full source code given, except:

```
    #### Fit a Gaussian over dense layer output ####

    # Means of Gaussian (10 classes)
    locs = tf.layers.dense(inputs=dense2, units=10, name="means")

    # Standard Deviations of Gaussian (10 classes)
    scales = tf.layers.dense(inputs=dense2, units=10,
                             name="std_devs", activation=tf.nn.softplus)

    # Parameterize the Gaussian
    dist = Normal(loc=locs, scale=scales)

    # Sample from Gaussian 1000 times
    num_sample = 1000
    logits = dist.sample([num_sample], name='logits')

    # Change shape of sampled logits
    logits = tf.transpose(logits, [1, 0, 2])

    # Replicate the true label 1000 times, once for each sample
    labels = tf.tile(labels[:, tf.newaxis], [1, num_sample])
```

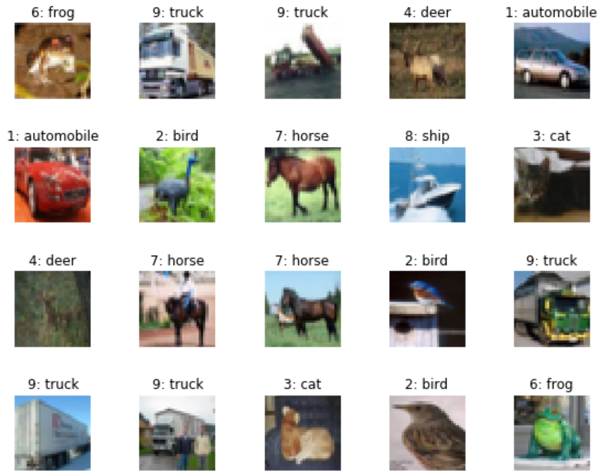
Reproduction

- Final version of reproduction uses:
  - Tensorflow
  - Keras
  - Lambda Layer
- Basis is an official Tensorflow MNIST tutorial



Results

- On MNIST dataset:
  - No increase in accuracy
  - Significant increase in training / testing time (4x longer)
  - The 1000 samples are basically just 1000 repeats of point estimate with tiny deviation
- On CIFAR-10 dataset:
  - Same results as MNIST



Conclusion

- Accuracy increase was not reproducable
- Maybe reproduction was wrong due to sparse information

It was not possible to gain the advantages of Bayesian NNs with reparametrisation.