A EXPERIMENT SETUP

A.1 MNIST, FASHION-MNIST and CIFAR-10 Experiment (section 5.1)

The SE-GP model is a vanilla Sparse Variational GP (SVGP) [Hensman et al., 2013] using a SE kernel defined directly on the images. For all datasets, we use the standard splits for the train and test set, as returned by the Keras dataset library. For comparison's sake, we set up TICK-GP and Conv-GP in as similar way as possible. They are both configured to have 1,000 inducing 5x5 patches, which are initialised using randomly picked patches from the training examples. We choose a SE kernel for the patch response function, and follow van der Wilk et al. [2017] in multiplying the patch response outputs with learned weights w_p before summation. Finally, we initialise the inducing patch locations $\ell(Z)$ of TICK-GP to random values in $[0, H] \times [0, W]$, and use a Matérn-3/2 kernel with the lengthscale initialised to 3 for the location kernel k_{loc} from eq. (2).

All GP models use a minibatch size of 128 and are trained using the Adam optimiser [Kingma and Ba, 2014] with a t^{-1} decaying learning rate, starting at 0.01. The models are run on a single GeForce GTX 1070 GPU until they converge.

Given that we are dealing with a multiclass classification problem, we use the softmax likelihood with 10 latent GPs. The softmax likelihood is not conjugate to the variational posterior, therefore we evaluate the predictive distribution using Monte Carlo estimates, $\frac{1}{K}\sum_k p(y_n \mid f^{(k)}(\cdot))$, where $f^{(k)}(\cdot) \sim q(f(\cdot))$. In our experiments we set $K = 5$.

A.2 Deep GPs Comparison (section 5.3)

We configure the deep convolutional GP models as identically as possible: each layer uses 384 inducing 5x5 patches (initialised using random patches from the training images), an identity Conv2D mean function for the hidden layers, and a SE kernel for the patch response function. The hidden layers for the L=2 and L=3 models are identical for both the deep Conv-GP and deep TICK-GP, because the translation insensitivity is added to the final layer only. We use a minibatch size of 32 for MNIST and, 64 for CIFAR. All models are optimised using Adam with an exponentially decaying learning rate, starting at 0.01 and decreasing every 50,000 optimisation steps by a factor of 4. We run all models for 300,000 iterations.

For the initialisation of the hidden layers' variational parameters, we follow Salimbeni and Deisenroth [2017] and set $\mathbf{m} = \mathbf{0}$ and $\mathbf{S} = \mathbf{I} \cdot 10^{-6}$. The zero mean and small covariance turn off the non-linear GP behaviour of the first layers, making them practically deterministic and completely determined by their identity mean function. In the final layer we set $\mathbf{m} = \mathbf{0}$ and $\mathbf{S} = \mathbf{I}$, as we do for the single-layer models in section 5.1. For the initialisation of the three-layer models we set the first and last layer to the trained values of the two-layered model, as was done in Blomqvist et al. [2019]. This is why we plot the optimisation curves for the three-layered models after the two-layer models in fig. 8.

B CNN Architectures

The Convolutional Neural Network (CNN) used in the classification experiments consists of two convolutional layers. The convolutional layers are configured to have 32 and 64 kernels respectively, a kernel size of 5x5, and a stride of 1. Both convolutional layers are followed by max pooling with strides and size equal to 2. The output of the second max pooling layer of size 1024 is fed into a fully connected layer with ReLU activation, the result of which is passed through a dropout layer with rate 0.5. The final fully connected layer has 10 units with softmax non-linearity. We initialised the convolutional and fully-connected weights by a truncated normal with standard deviation equal to 0.1. The bias weights were initialised to 0.1 constant. The CNN is trained using the Adam optimiser described in Kingma and Ba [2014] with a constant learning rate of 0.0001. We followed the architecture used in Keras.

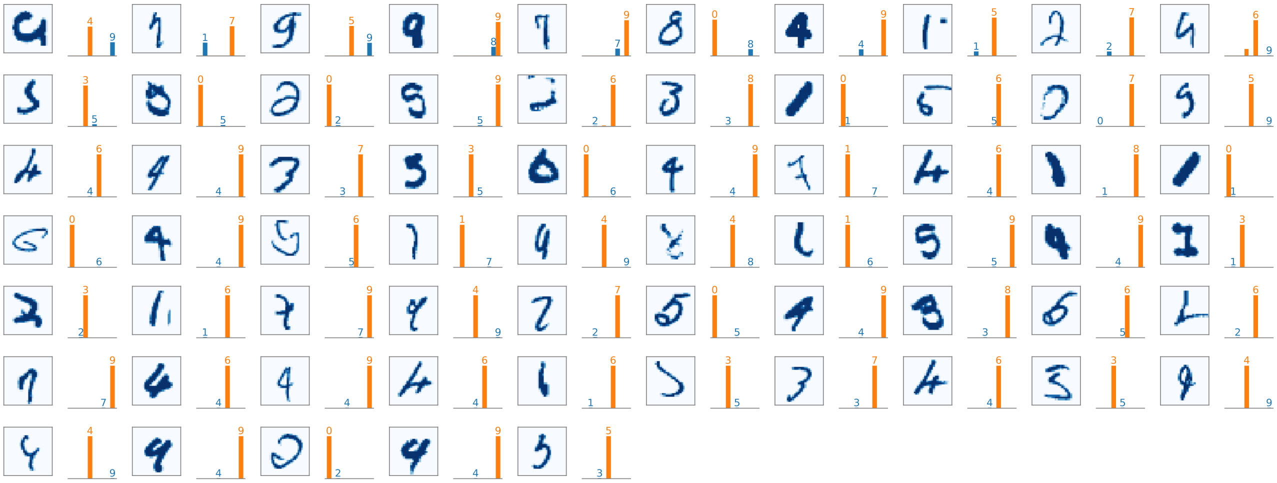

Figure 5: CNN model's prediction probabilities for misclassified MNIST images.

C Predictive Probabilities Of Misclassified MNIST Images

D Out-Of-Distribution Test

In this experiment we test the generalisation capacity of the models presented in section 5.1. In particular, we are interested in studying their behaviour when a distribution shift occurs on the test set. This is an important application, because most machine learning models will eventually be used in domains broader than their training dataset. It is therefore crucial that the models can detect this change of environment, and adjust their uncertainty levels so that appropriate actions can be taken.

The models in table 3 are trained on MNIST, but the reported metrics, error rate, and NLL are calculated for the Semeion digit dataset. The Semeion dataset [UCI] has 1,593 images of 16x16 pixels size. To be able to re-use MNIST trained models, we pad the Semeion images with zero pixels to match the MNIST size. The table shows that TICK-GP outperforms the CNN, and to a lesser extent the Conv-GP, in terms of NLL, and performs comparably to a CNN in terms of accuracy. In fig. 3 we show the predictive probability for the models for a few randomly selected *misclassified* images. The image clearly illustrates the fact that the CNN is making wrong predictions with a very high certainty, explaining the low NLL values.

Table 3: Results of Out-Of-Distribution test set experiment. The models are trained on MNIST digits and tested on the different Semeion digit dataset (lower is better).

metric	Conv-GP	CNN	TICK-GP
top-1 error	36.72	**14.44**	16.26
top-2 error	16.63	**5.27**	5.71
top-3 error	9.10	1.95	**1.76**
NLL full test set	1.027	2.115	**0.474**
NLL misclassified	2.221	14.614	**1.941**

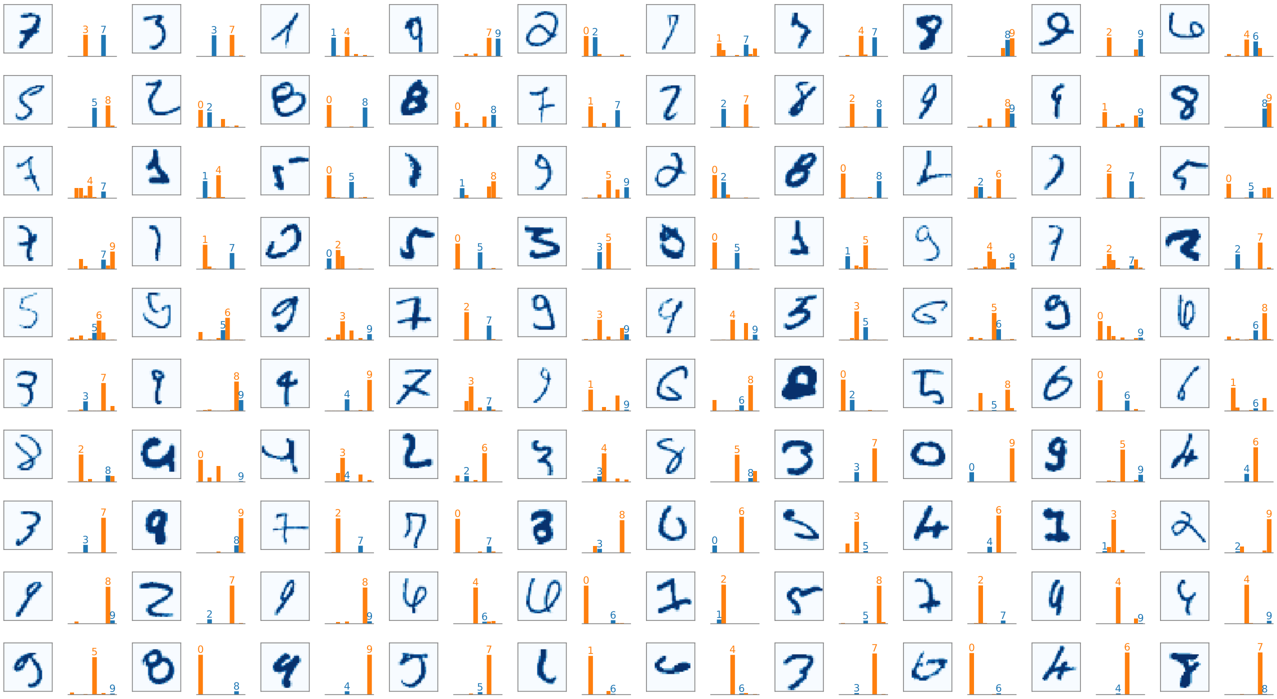

Figure 6: Conv-GP model's prediction probabilities for misclassified MNIST images.

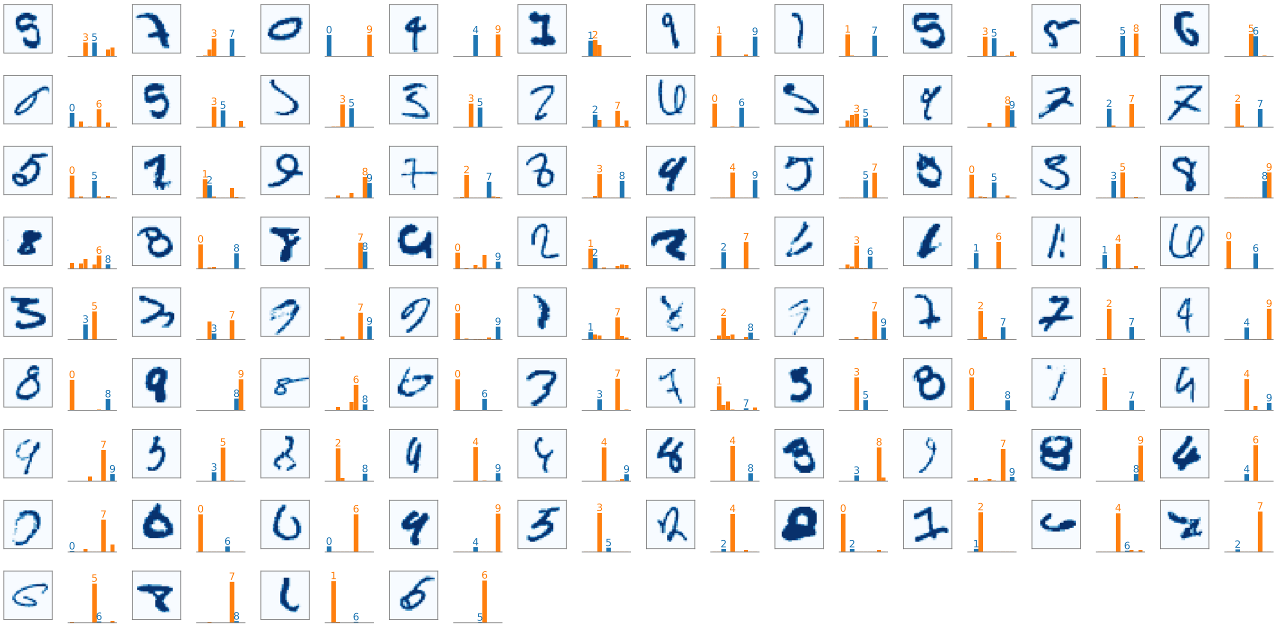

Figure 7: TICK-GP model's prediction probabilities for misclassified MNIST images.

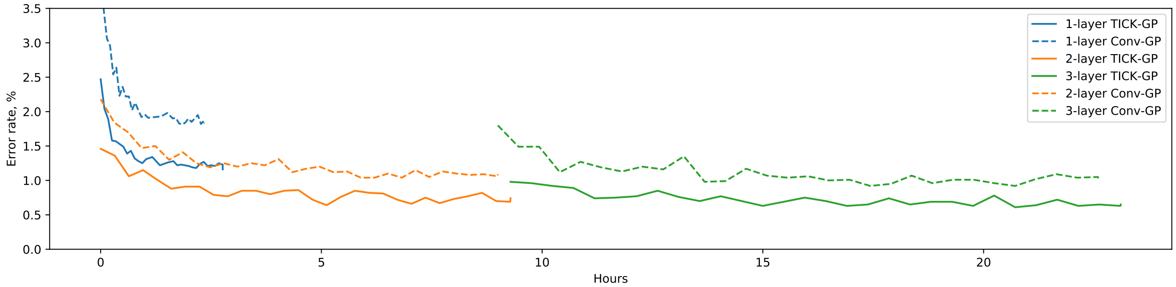

Figure 8: Deep convolutional GP error rate traces in function of optimisation time on the MNIST dataset. We plot TICK (solid) and Conv-GP (dashed) models, with one (blue), two (orange), and three (green) layers. All models ran for 300,000 iterations. The three-layered models are initialised with the trained values of the two-layered model.

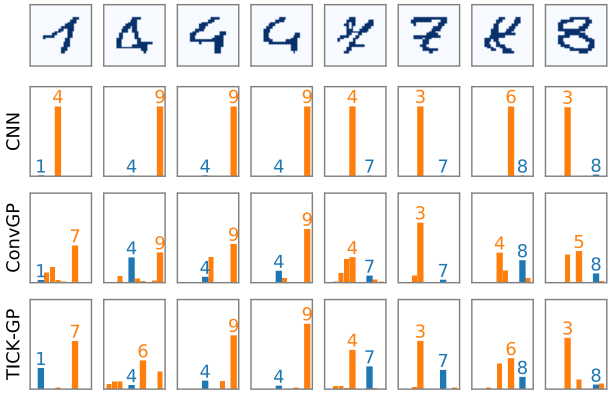

Figure 9: Prediction probabilities for eight randomly selected misclassified images (top row) form the Semeion dataset. The bars show the probabilities for each of the classes, 0 to 9. The largest orange bar is the class with highest probability and is thus used as a prediction from the model; the blue bar is the true class label.

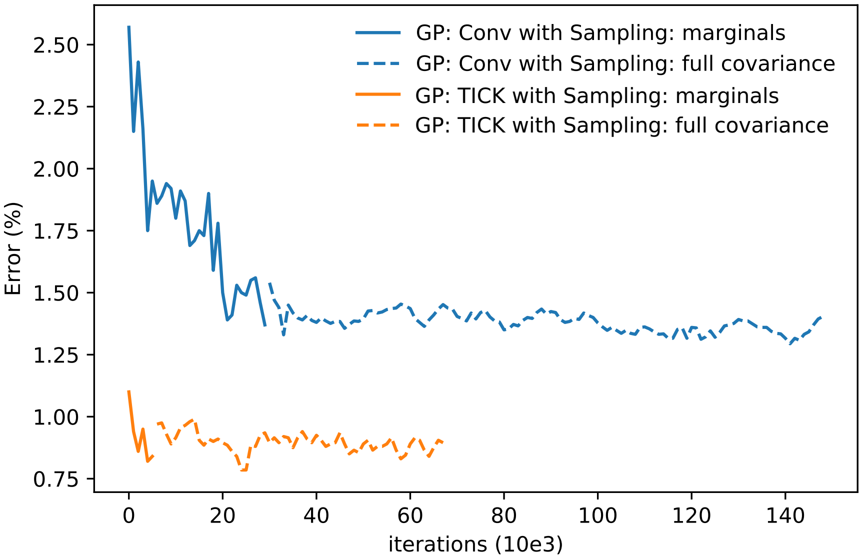

Figure 10: In this experiment we run two 2-layered deep convolutional models: one using the TICK kernel and another using the original convolutional kernel. We first optimise the models by sampling from only the marginals of the hidden layer's posterior GPs $q(f_\ell(\cdot))$, and then switch to sampling from the full covariance. We see the performance of the models slightly improving, but not enough to justify the added computational complexity. These are costly experiments, which took roughly 10 days to run.