Learning with a handful of pictures

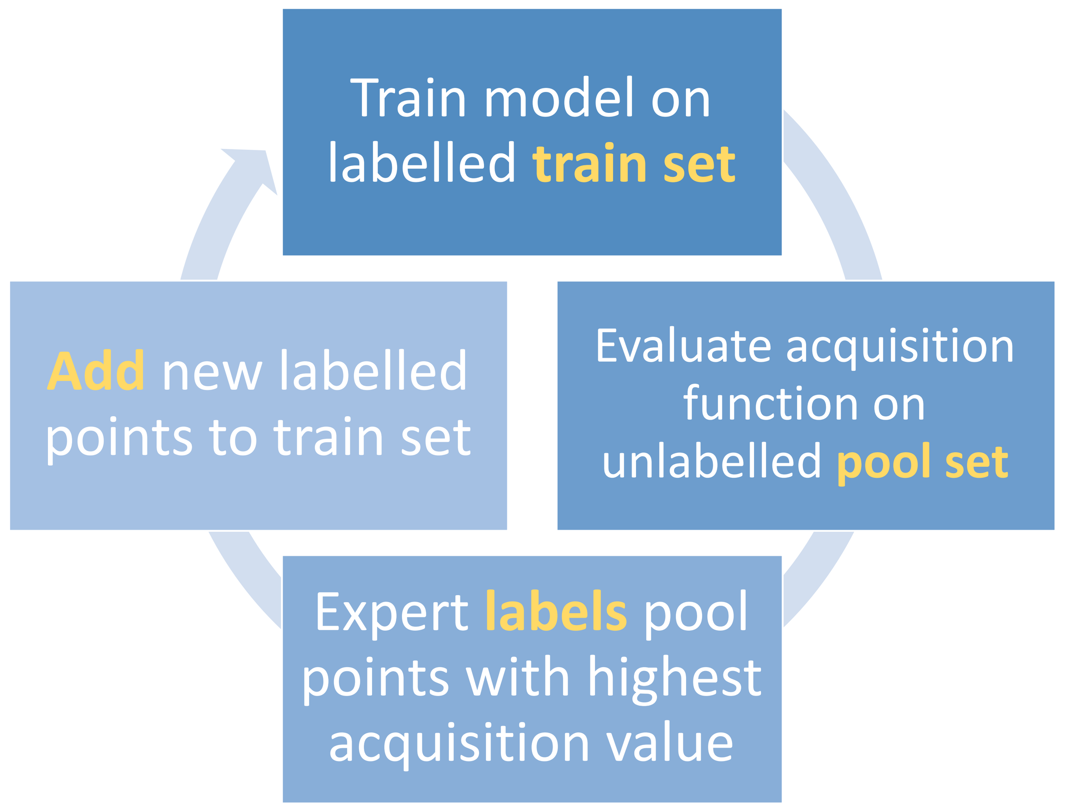
Using Bayesian CNNs for active learning of image data.

Yarin Gal (yg279), Riashat Islam (ri258), Zoubin Ghahramani (zg201), University of Cambridge @eng.cam.ac.uk

What, why, and how

- A big **challenge** in many applications is obtaining **labelled data**.
- **With active learning:** use a system that can **learn from small amounts of data**, and let the system **choose what data** it would like the user to label.
- For example, instead of labelling hundreds of dogs for a *dog breed classifier*, an ideal system should ask for a single label for each breed.
- Such systems make machine learning applicable to a wider class of problems.

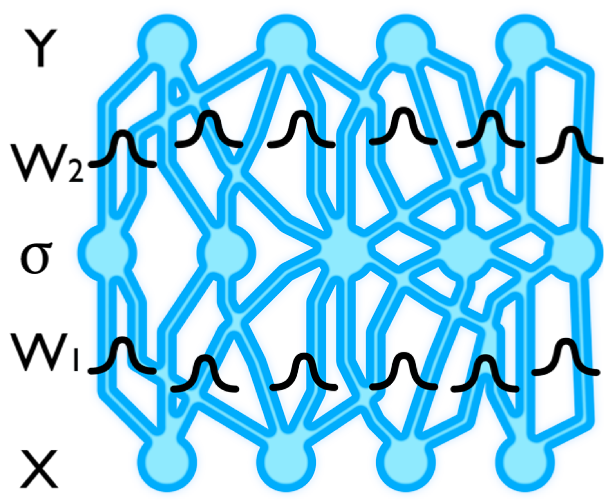
How:



Active learning forms an important pillar of machine learning, but active learning with image data is extremely challenging.

Bayesian CNNs

- Bayesian approaches to deep learning make handling **small data practical**
- Place a *prior* distribution over the CNN kernels, infer posterior distribution given data
- **Possess uncertainty information** for *acquisition functions*
- Bayesian CNN **approximate inference** can be done using various approximating distributions. For example a product of Bernoullis (implemented as dropout before each weight layer)



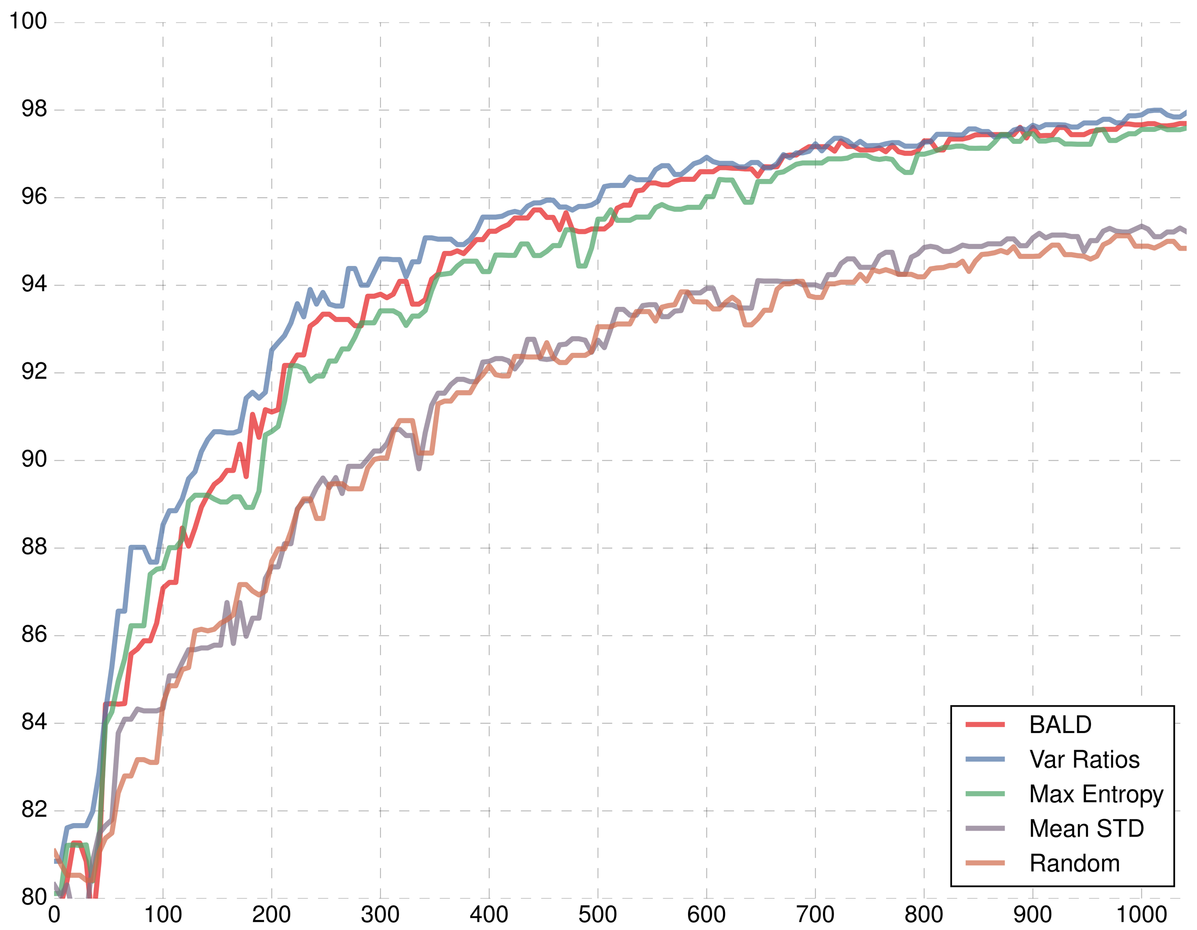
Acquisition functions

- Regression → look for images with high predictive variance
- But CNNs are often used for classification → other uncertainty measures needed
- Possible acquisition functions:

1. *Random* acquisition (baseline): $g(\mathbf{x}) = \frac{1}{N}$ with N pool points,
2. Maximise predictive entropy (*Max Entropy*, (Shannon, 1948))
$$\mathbb{H}[y|\mathbf{x}, \mathcal{D}_{\text{train}}] := -\sum_c p(y=c|\mathbf{x}, \mathcal{D}_{\text{train}}) \log p(y=c|\mathbf{x}, \mathcal{D}_{\text{train}}),$$
3. Maximise mutual information between predictions and model posterior (*BALD*, (Houlsby et al., 2011))
$$\mathbb{I}[y, \boldsymbol{\omega}|\mathbf{x}, \mathcal{D}_{\text{train}}] = \mathbb{H}[y|\mathbf{x}, \mathcal{D}_{\text{train}}] - \mathbb{E}_{p(\boldsymbol{\omega}|\mathcal{D}_{\text{train}})}\big[\mathbb{H}[y|\mathbf{x}, \boldsymbol{\omega}]\big]$$
with $\boldsymbol{\omega}$ the model parameters,
4. Maximise *Variation Ratios* (Freeman, 1965)
$$\text{variation-ratio}[\mathbf{x}] := 1 - \frac{f_{\mathbf{x}}}{T}$$
with $f_{\mathbf{x}} = \sum_t \mathbb{1}[y^t = c^*]$ and c^* being the mode of $\{y^t\}$,
5. Maximise *Mean STD* (Kendall et al., 2015)
$$\sigma(\mathbf{x}) = \frac{1}{C}\sum_c \sqrt{\mathbb{E}_{q(\boldsymbol{\omega})}[p(y=c|\mathbf{x}, \boldsymbol{\omega})^2] - \mathbb{E}_{q(\boldsymbol{\omega})}[p(y=c|\mathbf{x}, \boldsymbol{\omega})]^2}$$
averaged over all c classes $\mathbf{x}$ can take.

Active learning of MNIST

MNIST test accuracy as a function of # acquired images (up to 1000 images, using validation set size 100, and averaged over 3 repetitions):



Number of acquired images to get to model error of % on MNIST:

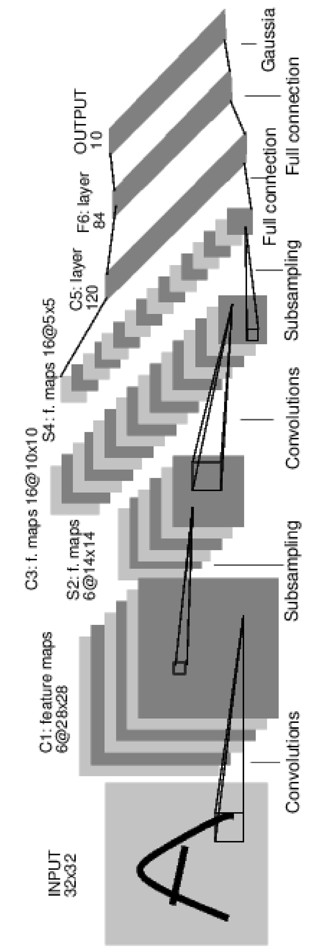
% error	BALD	Var Ratios	Max Entropy	Mean STD	Random
10%	145	120	165	230	255
5%	335	295	355	695	835

Comparison to semi-supervised learning

Setting: use 1000 labelled images for all techniques. Semi-supervised further has access to the remaining images with no labels. Active learning has access to **only** the 1000 acquired images. Following existing research we use a large val set of 5000.

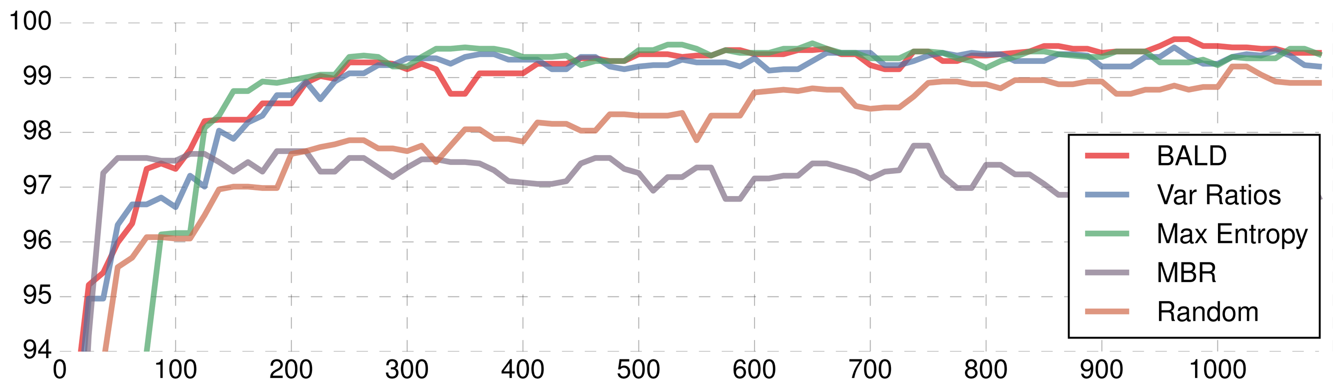
Technique	Test error
Semi-supervised:	
Semi-sup. Embedding (Weston et al., 2012)	5.73%
MTC (Rifai et al., 2011)	3.64%
Pseudo-label (Lee, 2013)	3.46%
AtlasRBF (Pitelis et al.,2014)	3.68%
DGN (Kingma et al., 2014)	2.40%
Virtual Adversarial (Miyato et al., 2015)	1.32%
Active learning with various acqs:	
BALD	1.80%
Var Ratios	1.64%
Max Entropy	1.74%
Random	4.66%

Test error on MNIST with 1000 labelled training samples



With active learning using simple Lenet and no unlabelled data!
(image source: LeCun et al. (1998))

Comparison to existing techniques for active learning of image data: Minimum Bayes risk – MBR (Zhu, Lafferty, Lafferty, 2003)



MNIST test accuracy (two digit classification) as a function of # acquired images

Acquisition function properties

Test accuracy as a function of # acquired images for various acquisition functions, using both a **Bayesian CNN** (red) and a **deterministic CNN** (blue):