MA8701

Active Learning

What data should I label?

Dr. Erlend Aune
Exabel & NTNU

Recap

Last lecture

Went through methodologies for dealing with little data.

N labeled datapoints : $\{x_i^L, \ldots, x_N^L\} = L$

$K \gg N$ unlabeled datapoints : $\{x_i^U, \ldots, x_K^U\} = U$

Transfer learning

Data Augmentation

Active Learning

Semi-supervised learning

One-shot learning

Last lecture

Do you have an idea about when the different techniques are appropriate?

N labeled datapoints : $\{x_i^L, \ldots, x_N^L\} = L$

$K \gg N$ unlabeled datapoints : $\{x_i^U, \ldots, x_K^U\} = U$

Transfer learning

Data Augmentation

Active Learning

Semi-supervised learning

One-shot learning

Suggestion 1: Active Learning

Take the bag-of-words example from Ben.

Extract a subset of size N (for example 1000), and leave a large “unlabeled” dataset

Use https://github.com/modAL-python/modAL to add samples from the unlabeled dataset.

Does it work better than sampling at random?

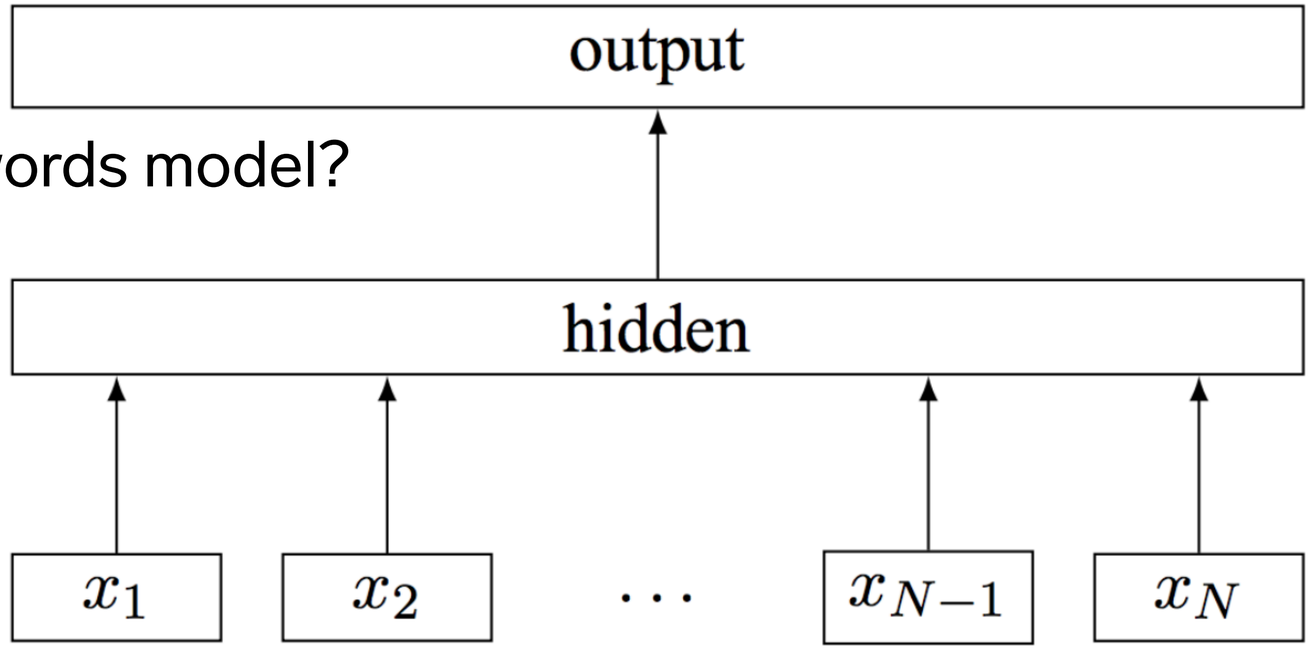
$$u_i = \sum_k P(y_i = k \,|\, x_i) \log P(y_i = k \,|\, x_i)$$

Suggestion 2: Transfer learning

Take russian word embeddings: https://rusvectores.org/en/models/.

Train a continuous bag of words model on this: That is use average of all word embeddings in a text, rather than one-hot encodings.

Does it work better than bag-of-words model?



Suggestion 3: Semi-supervised learning

Take a subset of 1500 images from MNIST (or CIFAR-10) - use 500 images as validation set

Use pseudo-labeling on full dataset to see how much you can improve validation error on the validation set.

Alternatively, modify code on https://github.com/Froskekongen/MA8701/blob/master/semisupervised/virtual_adv_training_baseline.py for semi-supervised learning

Suggestion 4: Siamese networks

Take a subset of 1500 images from MNIST (or CIFAR-10) - use 1000 images as validation set

Use siamese network to classify images into the different categories

Starting point: https://github.com/Goldesel23/Siamese-Networks-for-One-Shot-Learning

Note: This is a more challenging task than the others.

Exercises

Does the methodology for the exercises work? Why/why not?

Were there some issues?

Active Learning

How to label new data efficiently

Active Learning - two scenarios

Scenario 1: Add labels from ***U*** to ***L.*** Pool based Active learning

N labeled datapoints : $\{x_i^L, \ldots, x_N^L\} = L$

$K \gg N$ unlabeled datapoints : $\{x_i^U, \ldots, x_K^U\} = U$

Scenario 2

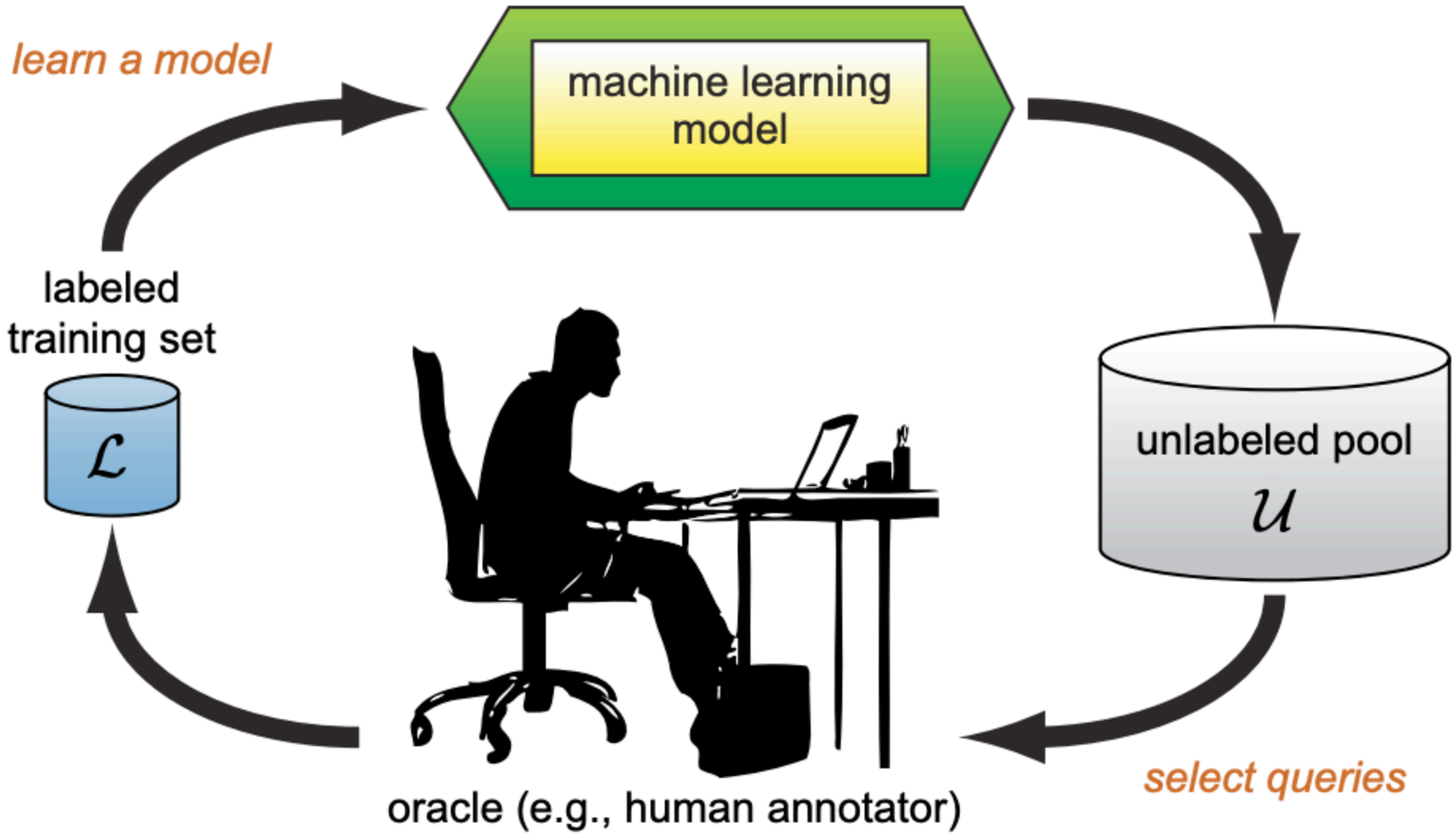
New data arrives in a stream:

- Make a prediction
- Ask an oracle

Pool Based Active Learning

Adding data from ***U*** to ***L***

Pool based AL



Pool based AL

- Let $\{x_j^i\} \subset f(U_i) \subset U_i$. f is then the query strategy
- Ask oracle for label for $\{x_j^i\} \forall j \in [1,k]$
- Let $L_i = L_{i-1} \cup \{x_j^i\}_{j=1}^k$
- Retrain model with $m_i = m(L_i)$

How does ***p(L$_i$)*** improve using that query strategy?

Uncertainty sampling

Main idea:

Let $\{\hat{y}_j\} = \{m(x_j)\}_i \forall x_j \in U_i$

Choose to label ***k*** samples: the ***y***'s where ***m*** is most uncertain its prediction.

Classification: Two uncertainty measures

Choose the label where the model is least confident about the preferred class

$$\hat{y}_j = \arg\max_y P(y \mid x_j)$$

$$x_j^i = \arg \max_{x \in U_i - \{x_k\}_{k=0}^{j-1}} (1 - P(\hat{y} \mid x))$$

Alternative: Margin sampling

$$\hat{\hat{y}}_j = \arg \max_{y \in Y - \hat{y}} P(y \mid x_j)$$

$$x^* = \arg\min_x (P(\hat{y} \mid x) - P(\hat{\hat{y}} \mid x))$$

Exercise: Which one performs best on MNIST?

Take a subset of 20 digits of the digits 8, 3 and 9 in MNIST.
Use margin sampling or least confident sampling in https://github.com/modAL-python/modAL,

https://modal-python.readthedocs.io/en/latest/content/apireference/uncertainty.html

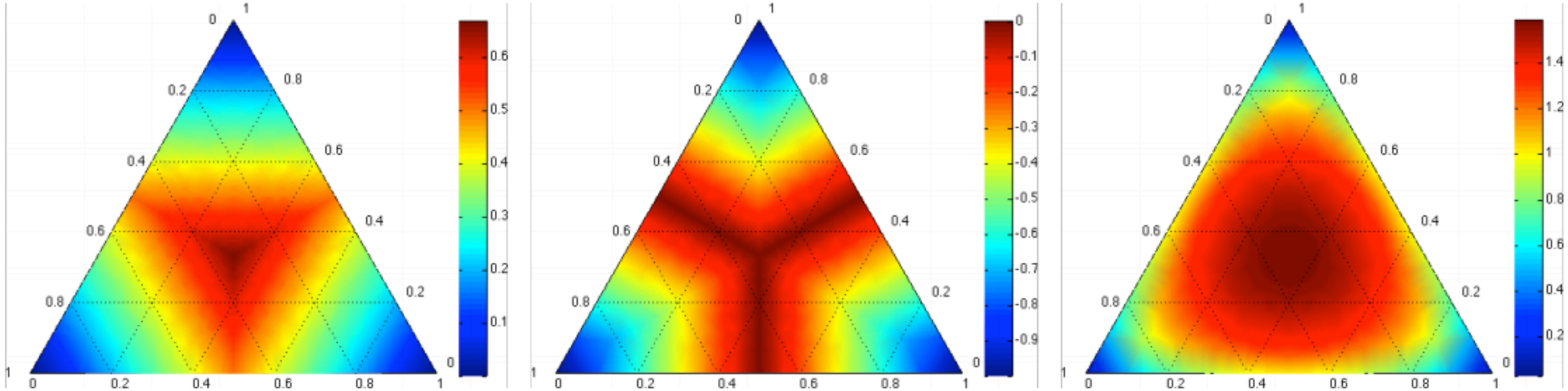
Which performs best?

Entropy sampling

Use a weighted sum of all classification scores

$$x_H^* = \underset{x}{\operatorname{argmax}} - \sum_i P_\theta(y_i|x) \log P_\theta(y_i|x),$$

This is a popular choice.



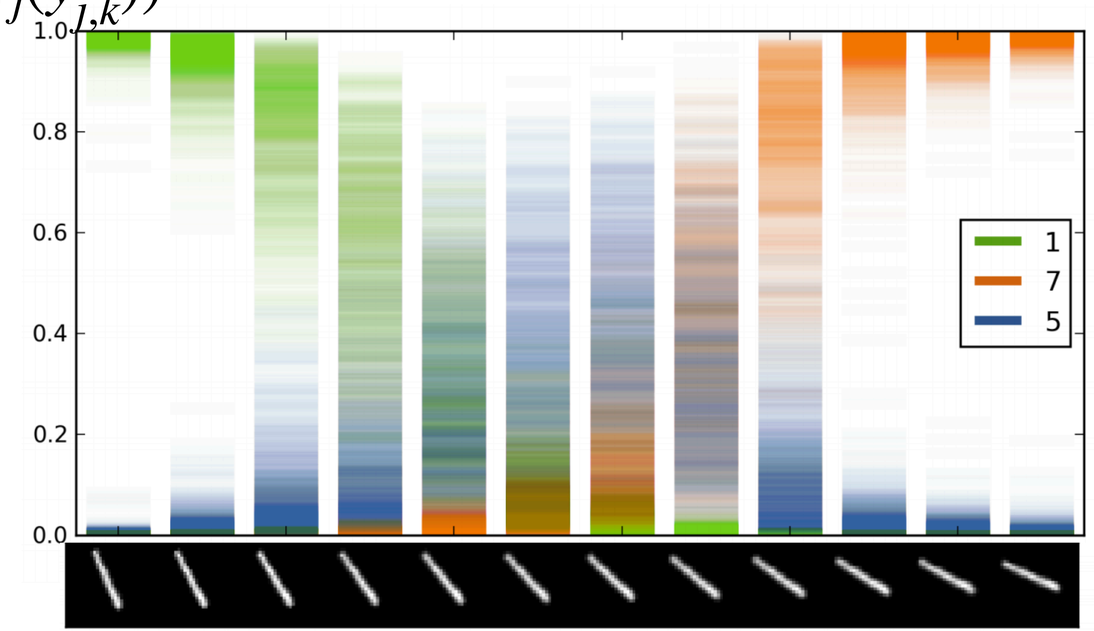
(a) least confident (b) margin (c) entropy

Dropout uncertainty

What if we use Dropout ***n*** times in the forward pass of a neural network? (https://arxiv.org/abs/1506.02142)

Uncertainty:

$$\text{Unc}(x_k) = \sum_{i=1}^{L} (\text{Var}_j(\hat{y}^i_{j,k}))$$



Query by Committee

Idea: Have ***n*** models, each performing predictions.

Models can be constructed in an arbitrary way (boosting, bagging, qualitative criteria, sampling)

Should be able to learn different aspects of data-label distribution.

Query by Committee

Metrics for model disagreement:

$$x^*_{VE} = \underset{x}{\operatorname{argmax}} - \sum_i \frac{V(y_i)}{C} \log \frac{V(y_i)}{C}$$

And:

$$x^*_{KL} = \underset{x}{\operatorname{argmax}} \frac{1}{C} \sum_{c=1}^{C} D(P_{\theta^{(c)}} \| P_{\mathcal{C}}),$$

$$D(P_{\theta^{(c)}} \| P_{\mathcal{C}}) = \sum_i P_{\theta^{(c)}}(y_i|x) \log \frac{P_{\theta^{(c)}}(y_i|x)}{P_{\mathcal{C}}(y_i|x)}.$$

Expected Gradient Length

Finding the ***x*** that changes the model the most is an alternative strategy.

$$x^*_{EGL} = \underset{x}{\operatorname{argmax}} \sum_i P_\theta(y_i|x) \left\| \nabla \ell_\theta(\mathcal{L} \cup \langle x, y_i \rangle) \right\|$$

$$\nabla l_\theta(\mathscr{L} \cup \langle x, y_i \rangle) \approx \nabla l_\theta(\langle x, y_i \rangle)$$

What gradients to look at?

It may be advantageous to look at gradients at only parts of the model.

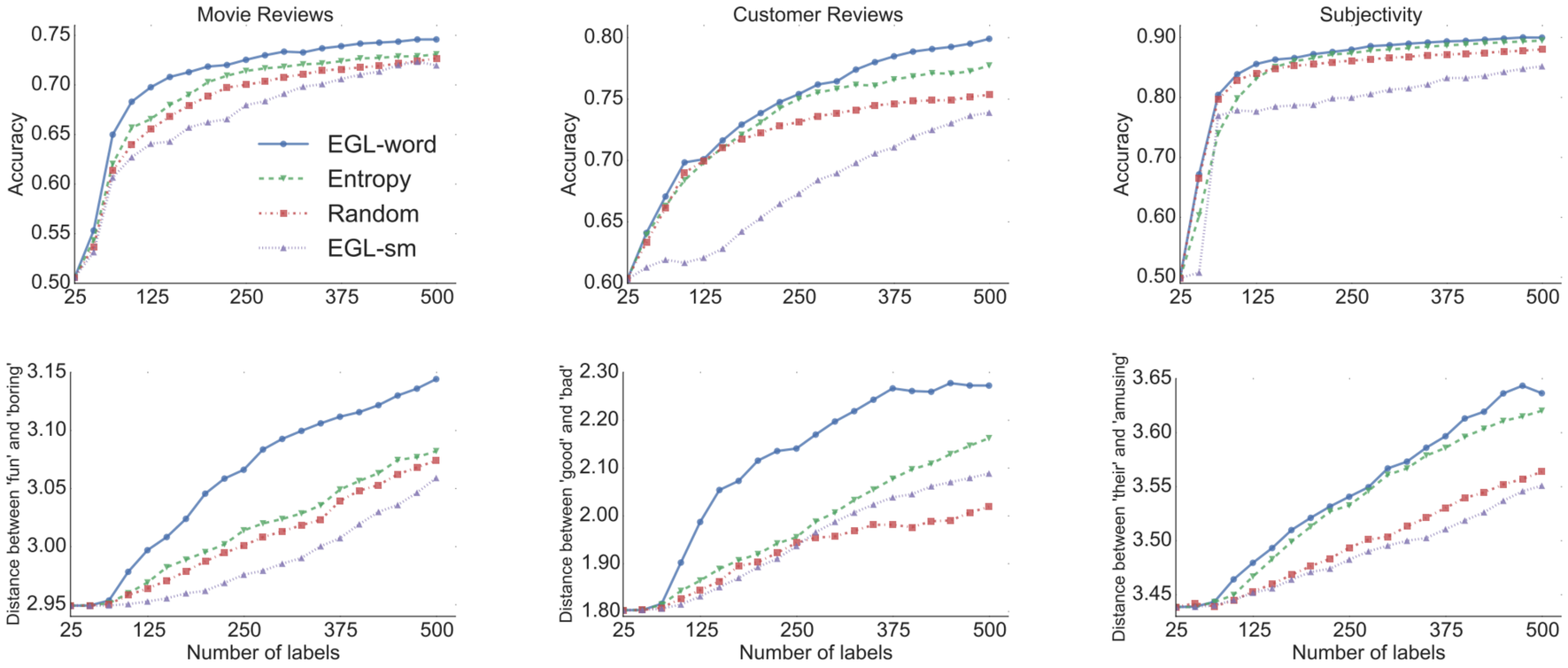
E.g. gradients of embedding layers in text problems.

$$x^*_{EGL} = \underset{x}{\operatorname{argmax}} \sum_i P_\theta(y_i|x) \left\| \nabla \ell_\theta(\mathcal{L} \cup \langle x, y_i \rangle) \right\|$$

$$\nabla l_\theta(\mathscr{L} \cup \langle x, y_i \rangle) \approx \nabla l_\theta(\langle x, y_i \rangle)$$

Expected Gradient Length (EGL)

Maximize EGL in (word) embeddings



Active Discriminative Text Representation Learning →

Getting gradients in Keras

Finding the ***x*** that changes the model the most is an alternative strategy.

$$x^*_{EGL} = \underset{x}{\operatorname{argmax}} \sum_i P_\theta(y_i|x) \left\| \nabla \ell_\theta(\mathcal{L} \cup \langle x, y_i \rangle) \right\|$$

$$\nabla l_\theta(\mathscr{L} \cup \langle x, y_i \rangle) \approx \nabla l_\theta(\langle x, y_i \rangle)$$

https://github.com/Froskekongen/MA8701/blob/master/gradients/keras_gradients.py

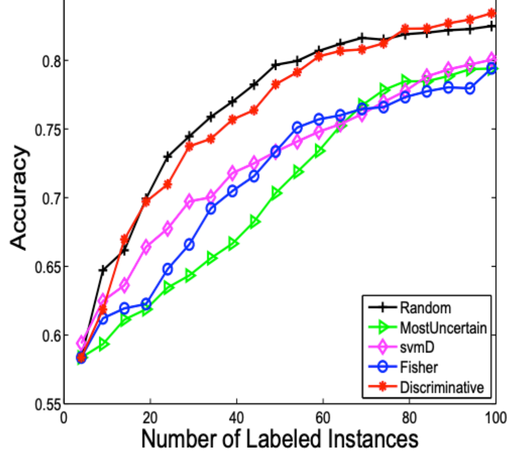
Density weighting

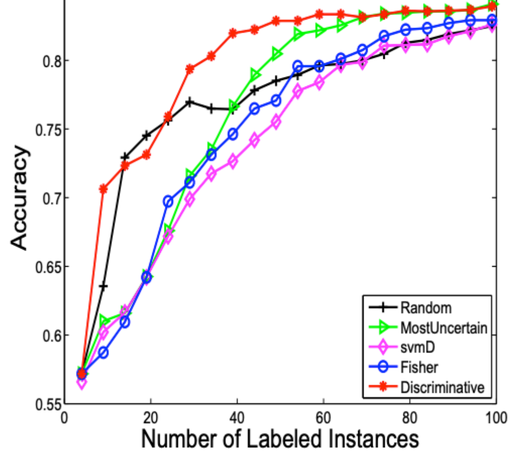
- Main idea: Augment query strategy by representing the data distribution.

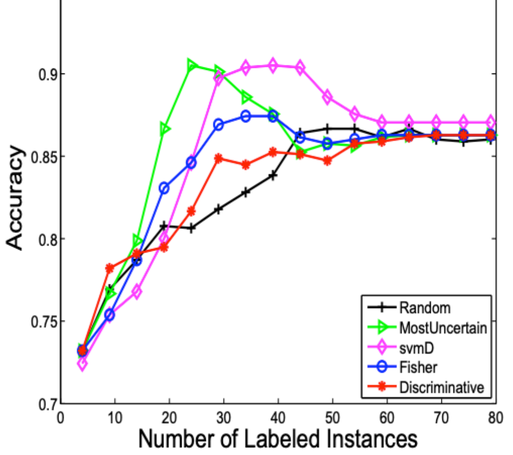
$$x_{ID}^{*} = \underset{x}{\operatorname{argmax}}\, \phi_A(x) \times \left(\frac{1}{U} \sum_{u=1}^{U} \operatorname{sim}(x, x^{(u)}) \right)^{\beta}$$

Other query strategies

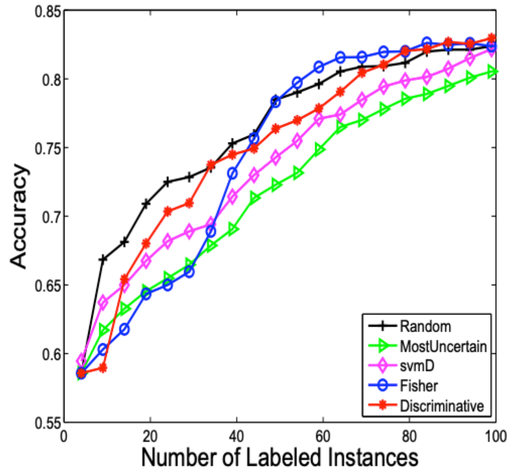
- Section 3.4-3.5 in the survey paper
- Expected Error Reduction
- Variance Reduction



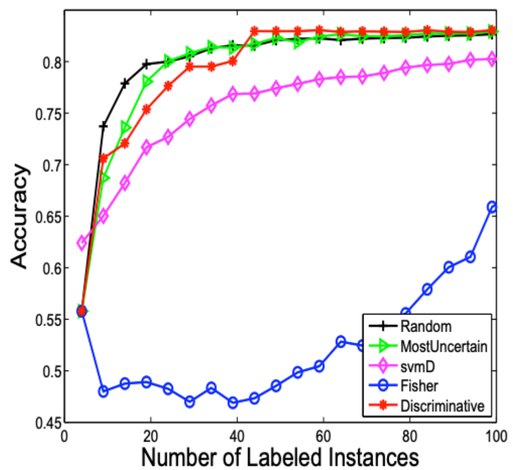




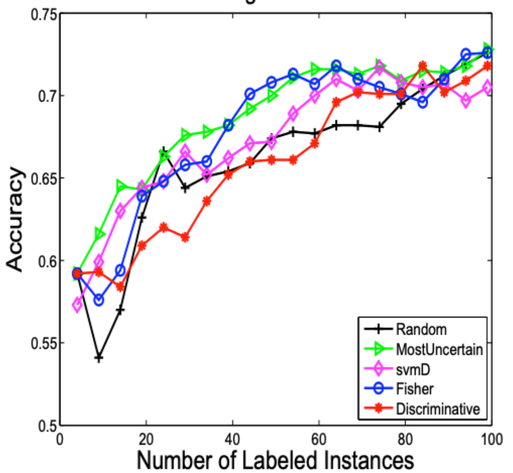
crx



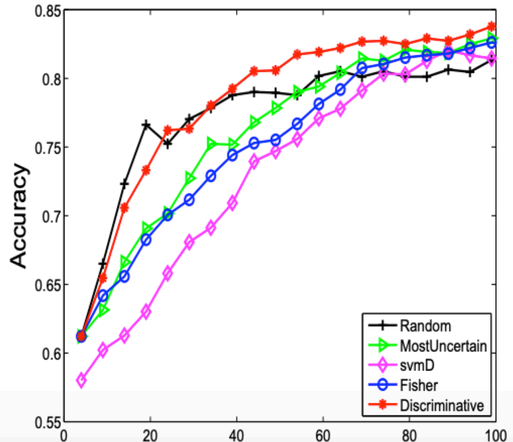
flare



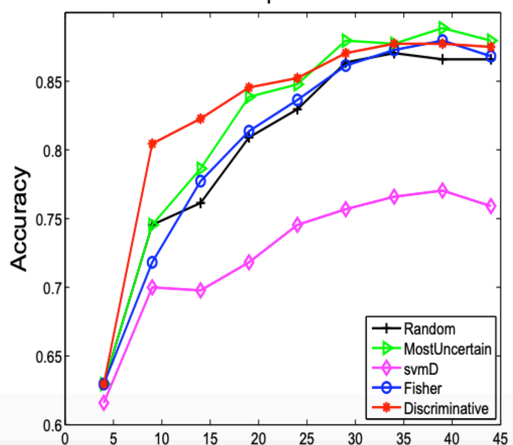
glass2



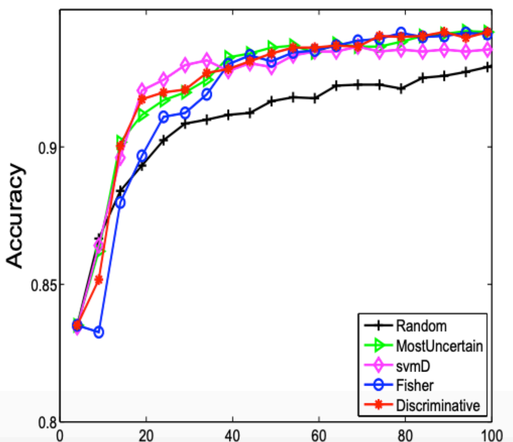
heart



hepatitis



vote



Discussion

What are the main issues with the strategies we have looked at?

- Training time?
- Data set sizes?

Are there similarities to the techniques we discussed last lecture?

Does it make sense to always query the most uncertain sample?

Main challenges

- Querying outliers
- Intrinsic uncertainty in data
- Adding one label at the time before retraining

Alternative formulations

Querying for features

- Assume that we have icomplete data.

x_i^f, f may be missing for many data points .

Goal: Choose the missing ***f*** that improves the model the most.

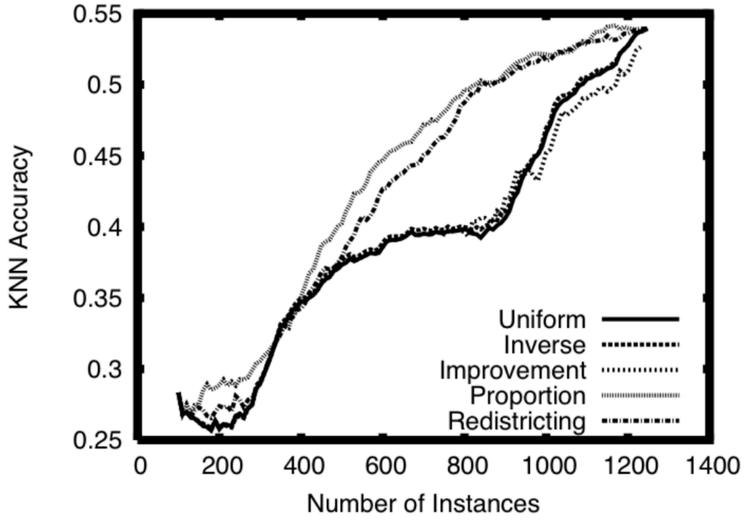
Still possible to use query strategies, but the cost of obtaining ***f*** should be addressed in the strategy.

Connected to Value of Information

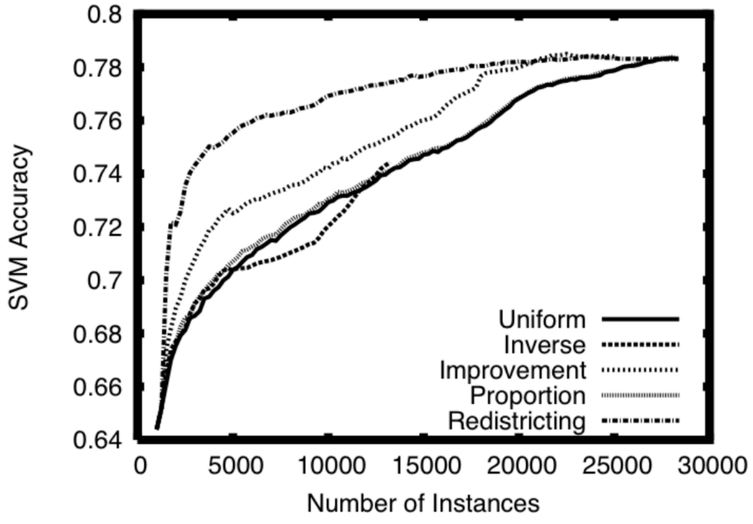
Active Class Sampling

Goal: Sampling the class that improves the model the most - without ***U***.*https://link.springer.com/chapter/10.1007/978-3-540-74958-5_63*

Example: $P_r[c] := max(0, \frac{currAcc[c] - lastAcc[c]}{\sum_{i=1}^{|classes|} currAcc[i] - lastAcc[i]} * b[r])$



(a) Nose



(b) Land Cover

Batch-mode AL

Get more than one sample in query.

Retraining after one-sample query is potentially very different from querying for a large batch using the same criterion.

Question: How to induce diversity in a batch?

- Explicit modeling: https://papers.nips.cc/paper/3295-discriminative-batch-mode-active-learning.pdf

- Heuristics

Diverse mini-batch AL

Simple strategy in: https://arxiv.org/abs/1901.05954

Let s_i be the query score for sample i.

Minimize k-means objective:

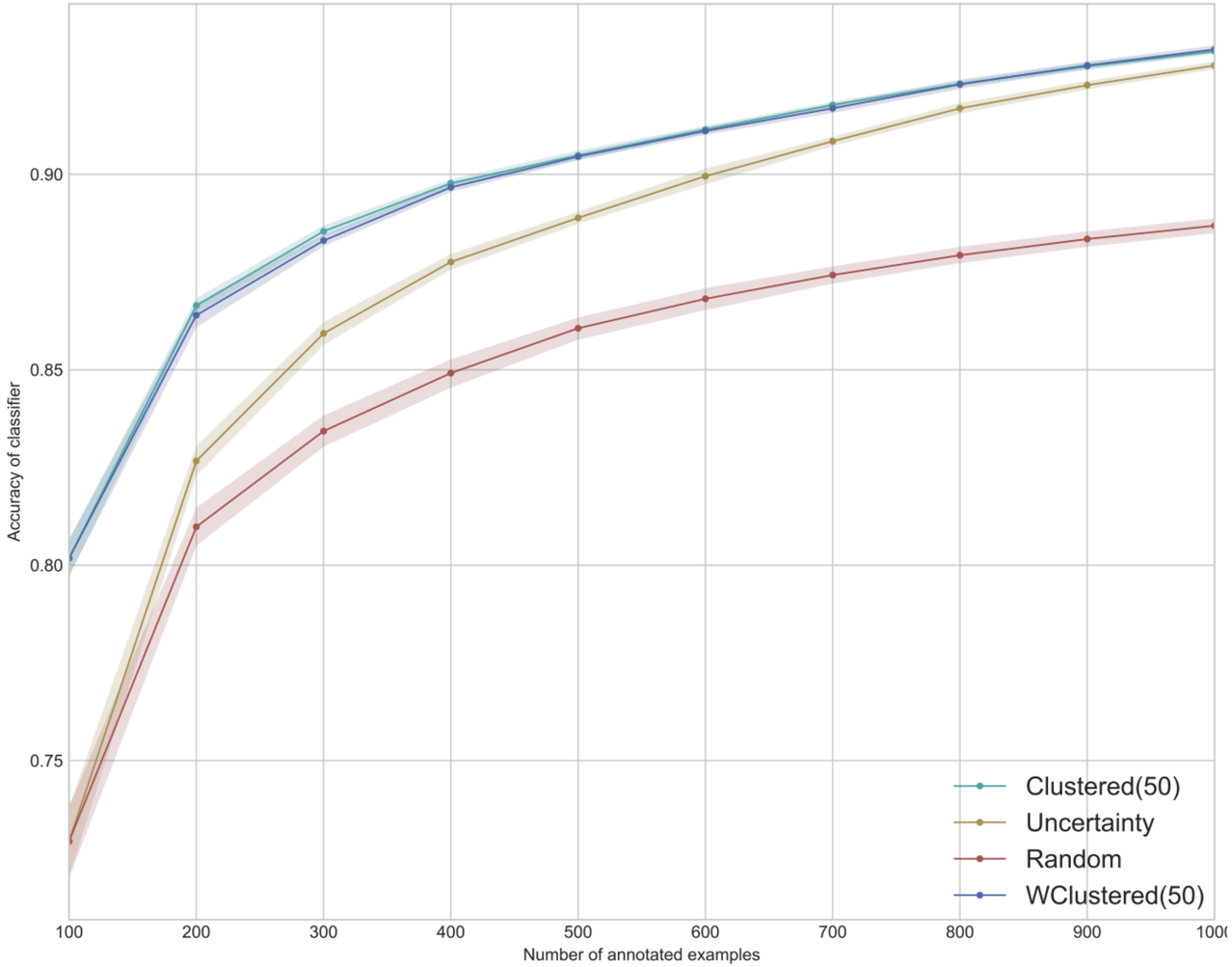
$$\sum_{x_i \in \mathcal{X}^U} z_{i,k} s_i \|x_i - \mu_k\|^2 \to \min$$

Algorithm 1 Diverse mini-Batch Active Learning (DBAL)

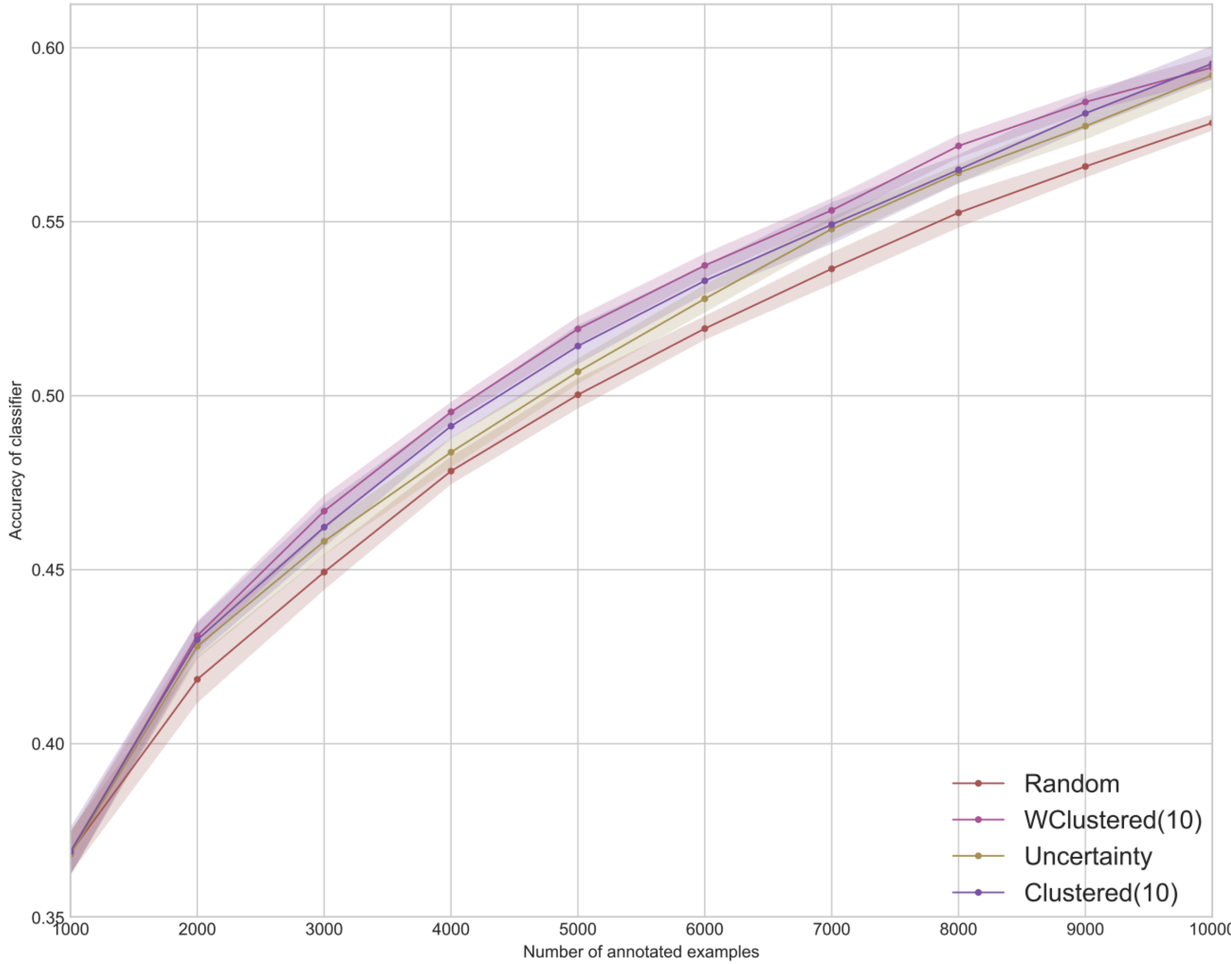
Input: dataset of examples x_i, budget B, batch-size k, pre-filter factor β
Select first k examples randomly, obtain labels for these examples
repeat
  Train classifier on all the examples selected so far
  Get informativeness for every unlabeled example
  Prefilter to top βk informative examples
  Cluster βk examples to k clusters with (weighted) K-means
  Select k different examples closest to the cluster centers, obtain labels for these examples
until Budget B is exhausted

Diverse mini-batch AL

MNIST



CIFAR-10



Observation noise

The observations (labels) are often noisy.

Can lead to information collapse. What happens if there is more noise in parts of the data space?

However: Easy to simulate effects! https://modal-python.readthedocs.io/en/latest/content/examples/active_regression.html

Labeling noise/labeling cost

More accurate equipment: More precise labeling, while more expensive.

Treatment: One medicine is more costly than another.

Human annotators have different experience/knowledge. Annotation fatigue is also a real issue.

Changing the model

AL -> Biased training data

Does this biased data improve model performance if the model class is changed?

Use ensemble of model and combine query strategy for the ensemble?

- Alternate between the models for selection
- Aggregate uncertainties
- Use different features for each model?

Recent approaches in Deep Learning

AL for convnets: A core set approach

Core set: Is there a subset of ***S*** that can make the model perform (almost) as good as using all of ***S***?

$$\min_{\mathbf{s}^1:|\mathbf{s}^1|\leq b} \left| \frac{1}{n} \sum_{i\in[n]} l(\mathbf{x}_i, y_i; A_{\mathbf{s}^0\cup\mathbf{s}^1}) - \frac{1}{|\mathbf{s}^0 + \mathbf{s}^1|} \sum_{j\in\mathbf{s}^0\cup\mathbf{s}^1} l(\mathbf{x}_j, y_j; A_{\mathbf{s}^0\cup\mathbf{s}^1}) \right|$$

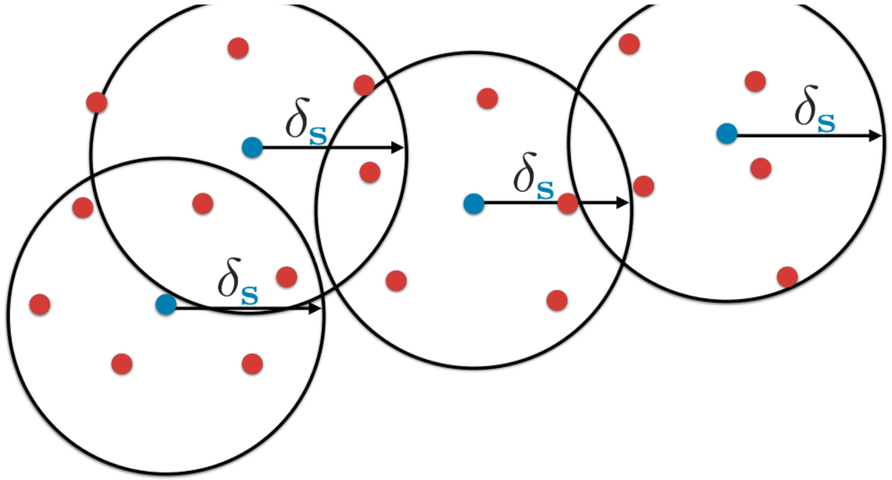


Figure 1: **Visualization of the Theorem 1**. Consider the set of selected points $\mathbf{s}$ and the points in the remainder of the dataset $[n] \setminus \mathbf{s}$, our results shows that if $\mathbf{s}$ is the $\delta_{\mathbf{s}}$ cover of the dataset, $\left|\frac{1}{n}\sum_{i\in[n]} l(\mathbf{x}_i, y_i, A_{\mathbf{s}}) - \frac{1}{|\mathbf{s}|}\sum_{j\in\mathbf{s}} l(\mathbf{x}_j, y_j; A_{\mathbf{s}})\right| \leq \mathcal{O}(\delta_{\mathbf{s}}) + \mathcal{O}\left(\sqrt{\frac{1}{n}}\right)$

https://arxiv.org/abs/1708.00489

AL for convnets: A core set approach

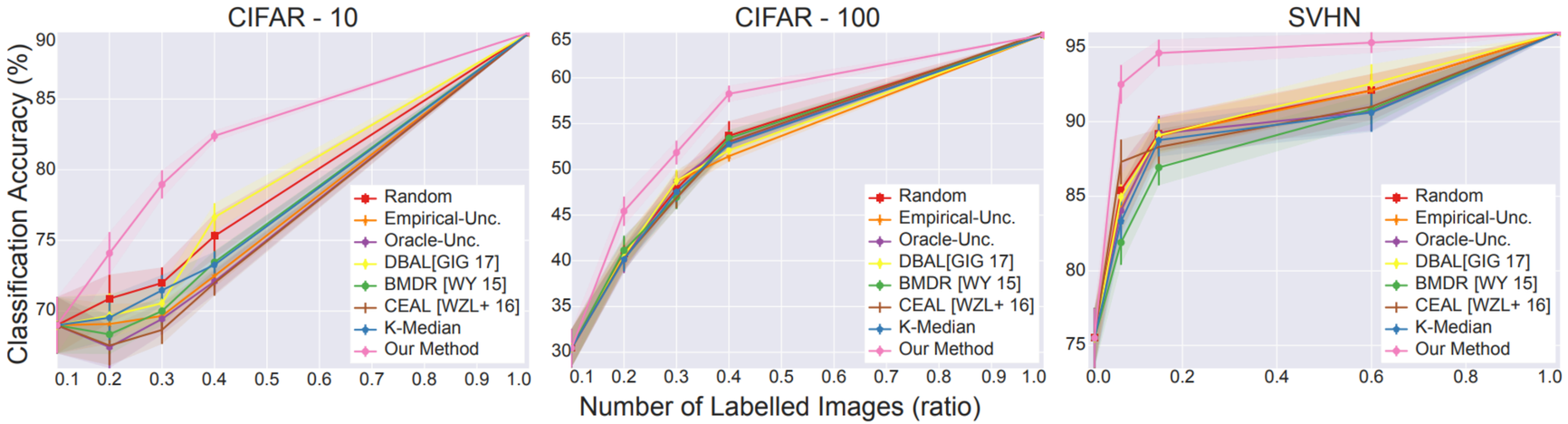


Figure 3: Results on Active Learning for Weakly-Supervised Model (error bars are std-dev)

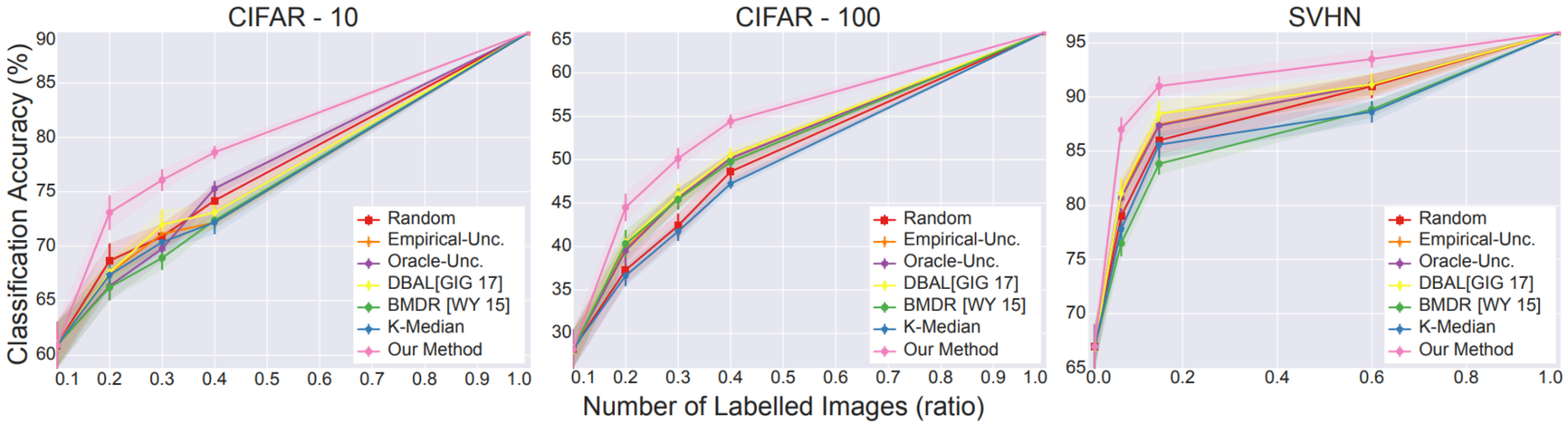


Figure 4: Results on Active Learning for Fully-Supervised Model (error bars are std-dev)

Deep Bayesian AL with Image Data

Bayesian NN: Prior on weights.

Approximate posterior using Dropout or other regularization.

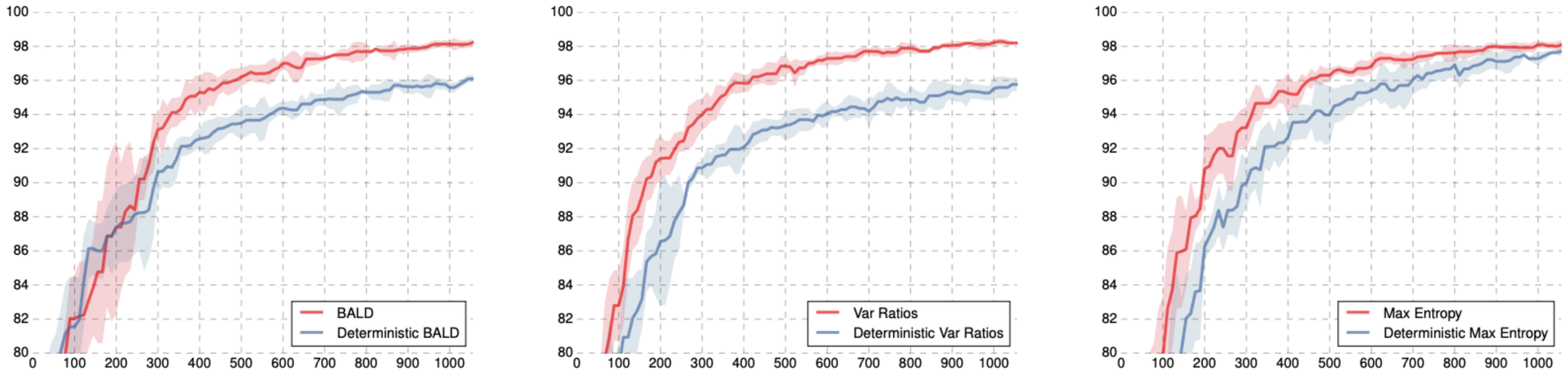
Main:

- Using Bayesian posterior approximation yields better query strategy

https://arxiv.org/abs/1703.02910

Deep Bayesian AL with Image Data

https://github.com/Riashat/Deep-Bayesian-Active-Learning



(a) BALD (b) Var Ratios (c) Max Entropy

Figure 2. Test accuracy as a function of number of acquired images for various acquisition functions, using both a **Bayesian CNN (red) and a deterministic CNN (blue)**.

AL is closely related to uncertainty

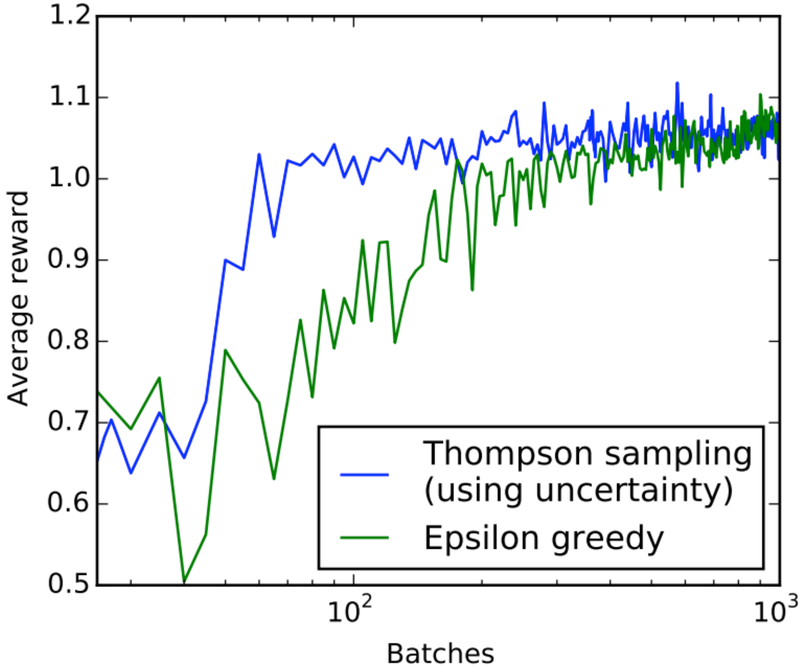
- A good read is Uncertainty in Deep Learning: http://mlg.eng.cam.ac.uk/yarin/thesis/thesis.pdf

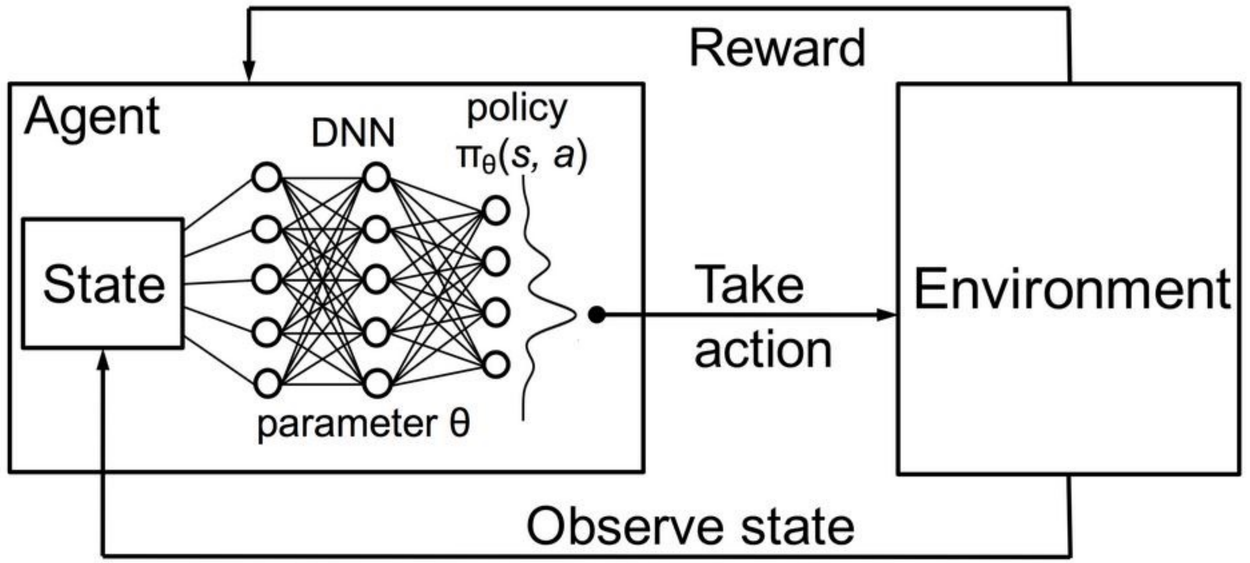
- In addition AL closely resembles Exploration in Reinforcement Learning. The question then is: How should one explore?

- In deep learning - more parameters seem to give better uncertainty estimates. “Larger uncertainties away from the data.”

Example: Exploration in RL

- Thompson sampling: Use Dropout when choosing an action
- In Experience Replay - use stochastic forward pass as well.
- Outperms epsilon-greedy strategy on many tasks





Example: Exploration in RL

- Alternative: Choose when to explore?
- When is the model uncertain about actions?

Streaming data is coming in

- Alternative 1: Model classifies data
- Alternative 2: Model is too uncertain to classify, and asks an oracle

Can we use the query strategies we have mentioned so far?

If so, how?

5 min discussion/break

Streaming data is coming in

- Add points after discussion

Meta-learning

A challenge in AL

- Many heuristics
- How to choose the right one for the problem at hand?
- Are we sure there aren’t better choices?

Alternative: Meta learning

Can we learn the heuristic of choice?

Meta-learning

Technique of choice: Reinforcement learning (e.g. https://arxiv.org/abs/1806.04798)

Goal: Select the point to query

Reward: Increase in performance using the point of choice

Transferable Active Learning Policies by Deep Reinforcement Learning

https://arxiv.org/abs/1806.04798

Z_U, Z_L matrices of unlabeled/labeled data

$$(\boldsymbol{W}_e)_j = \Psi\Big([\boldsymbol{e}_j^1(\boldsymbol{Z}_u^T), \boldsymbol{e}_j^1(\boldsymbol{Z}_l^T), \boldsymbol{e}_j^2([\boldsymbol{Z}_u^T, \boldsymbol{Z}_l^T], f_t)]\Big)$$

$$\pi(a_i|s_t) \propto \exp^{\Phi_{\theta_p}(\boldsymbol{W}_e^T \boldsymbol{z}_i)}$$

Ψ is a meta-network

e_j^k encodes feature/label histograms

Result: Slightly better than entropy sampling.

One-Shot learning - recap

Given a sequence of observations

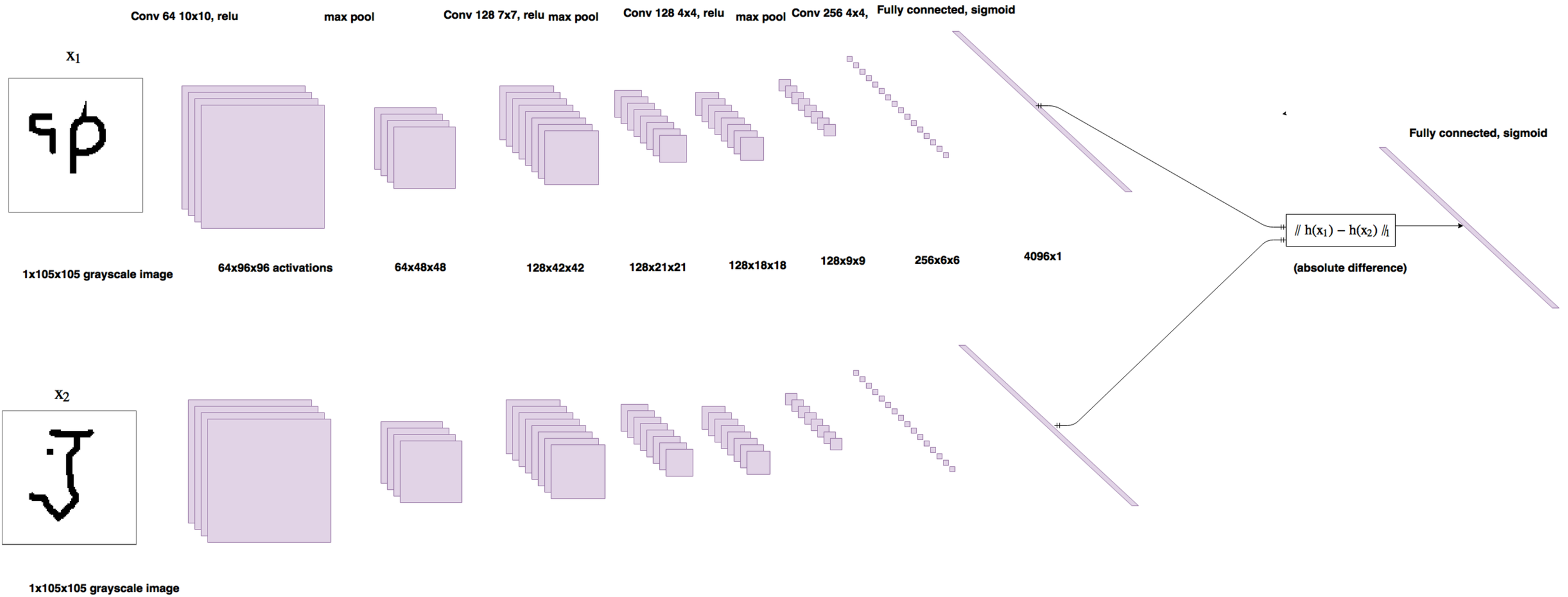
$$((x_1, y_{c_1}), (x_2, y_{c_2}), \ldots, (x_k, y_{c_k}))$$

y_{c_1} has been seen once.
Now we observe

$$(x_{k+1}, y_{c_1})$$

The model should recognize that x_{k+1}, x_1 has the same label.

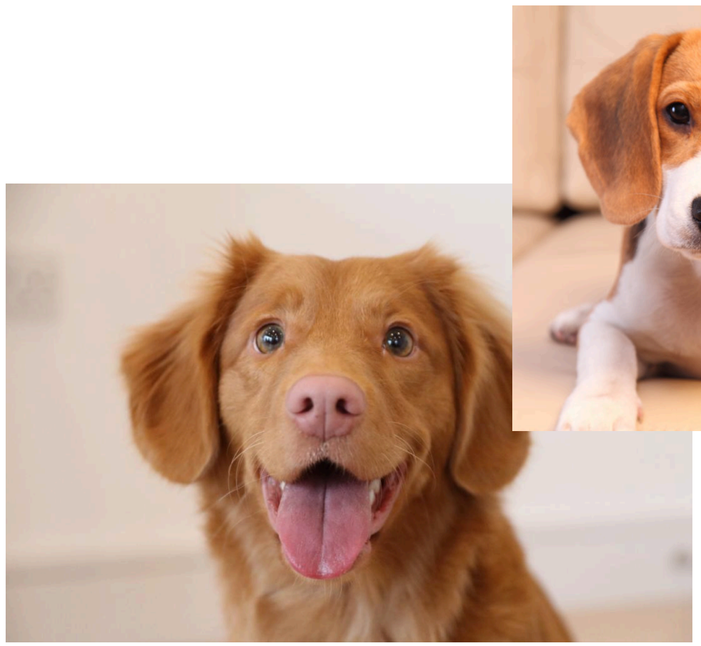
Siamese network recap

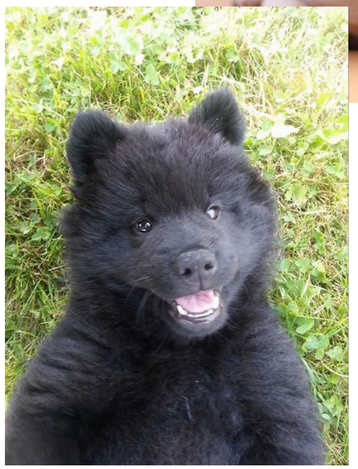


Siamese networks - issues

Must choose good contrastive examples.

Need heuristics for “what is close”



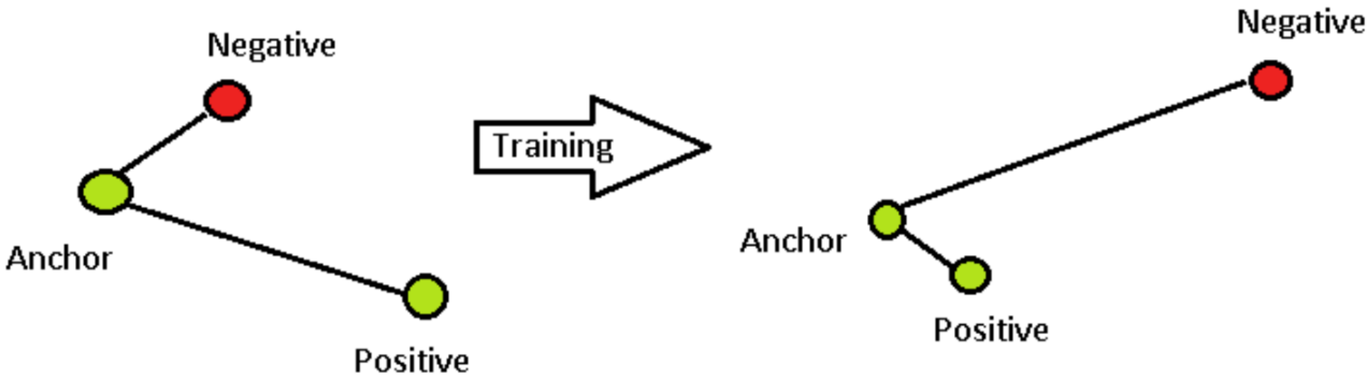


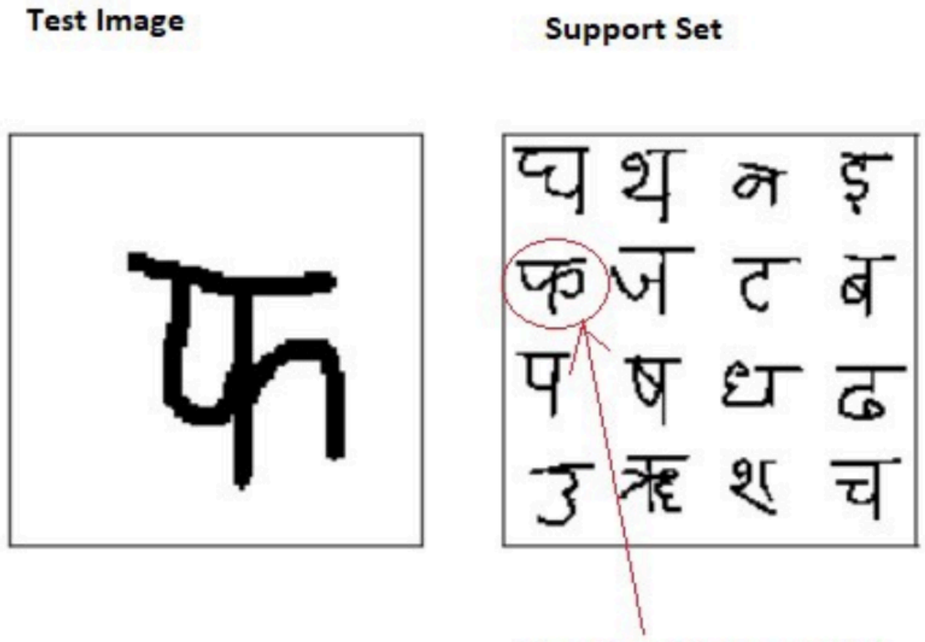
Siamese networks

Challenge

Choose three good data points: Anchor, positive and negative examples.

https://arxiv.org/abs/1503.03832





Metalearning: Learning an active learner

Consider the following scenario:

- An agent may choose whether to request a label or not
- The reward for the agent is $r_t = \begin{cases} R_{req}, & \text{if a label is requested} \\ R_{cor}, & \text{if predicting and } \hat{y}_t = y_t \\ R_{inc}, & \text{if predicting and } \hat{y}_t \neq y_t \end{cases}$
- Agent has access to representation of an example

	Accuracy (%)	Requests (%)
Supervised	91.0	100.0
RL	75.9	7.2
RL Prediction	81.8	**7.2**
RL Prediction ($R_{inc} = -5$)	86.4	31.8
RL Prediction ($R_{inc} = -10$)	89.3	45.6
RL Prediction ($R_{inc} = -20$)	**92.8**	60.6

Active One-shot Learning →

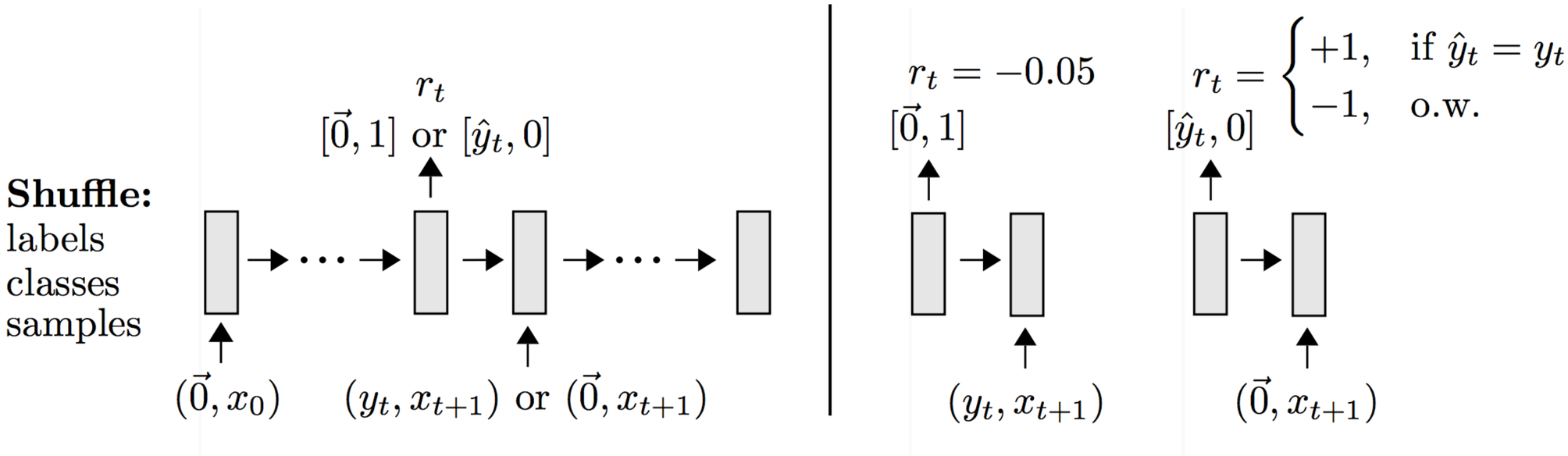
One shot learning

Goal: When a class has been seen once, it should not need to see more examples in same class in order to classify correctly.

	Accuracy (%)	Requests (%)
Supervised	91.0	100.0
RL	75.9	7.2
RL Prediction	81.8	**7.2**
RL Prediction ($R_{inc} = -5$)	86.4	31.8
RL Prediction ($R_{inc} = -10$)	89.3	45.6
RL Prediction ($R_{inc} = -20$)	**92.8**	60.6

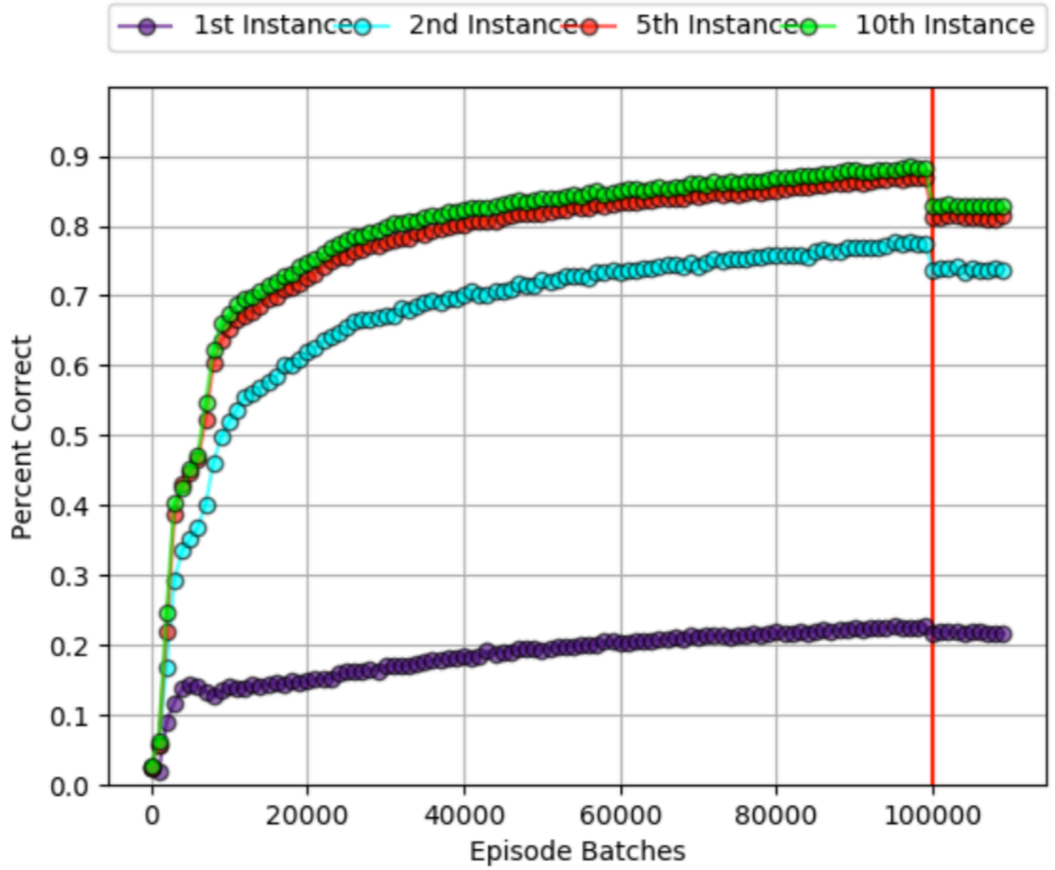
Active One-shot Learning →

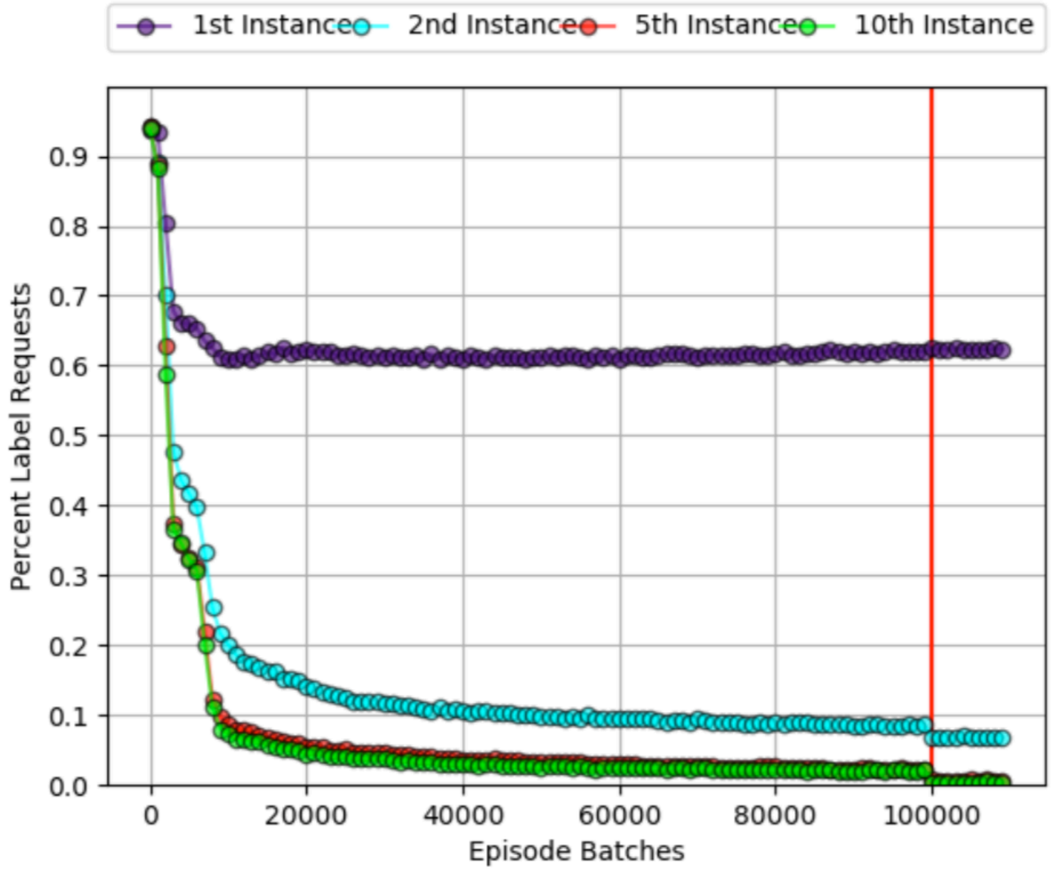
Task structure



(a) Task structure (b) Request label (c) Predict label

Performance of one-shot learning over episodes





Performance of one-shot learning within episodes

Combining small-data techniques

AL and semi-supervised learning

- Use both information from ***U*** for strengthening the model AND choosing samples.

Exercise: Combine pseudo-labeling with entropy strategy: Pseudo label the least-entropic labels, and query the most entropic. (Subsample 200 imgs from CIFAR-10 and MNIST as a starting point for ***L***)

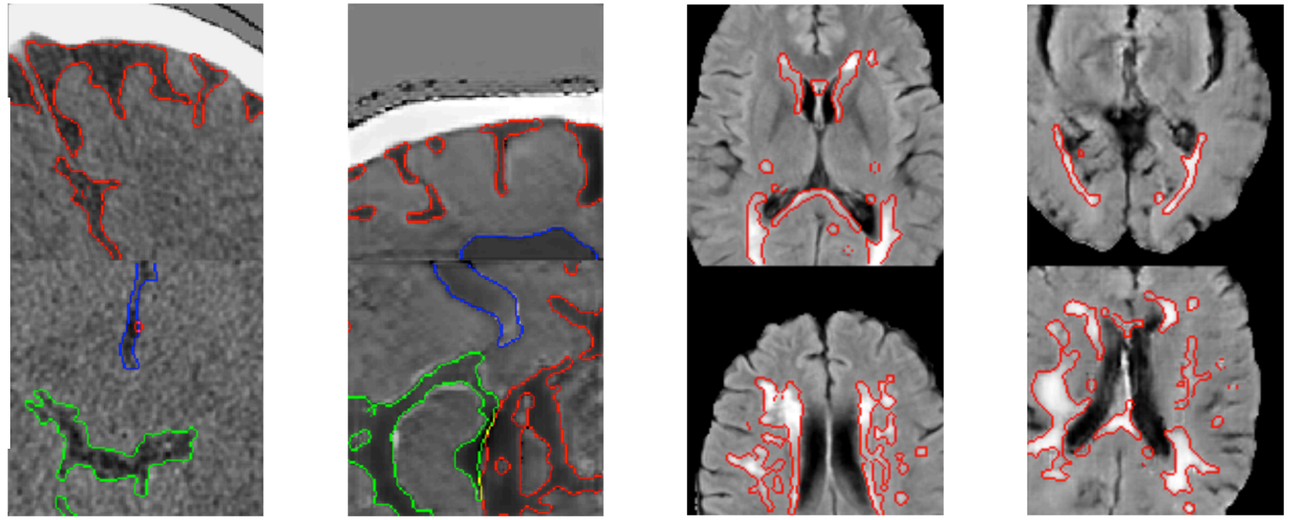
Does this strategy work?

AL data augmentation

- Recent research tries using GANs for data augmentation (https://arxiv.org/abs/1810.10863)

- Can we combine AL with such data augmentation to increase robustness? What synthetic data should be labeled? What if we also have a pool of unlabeled data?

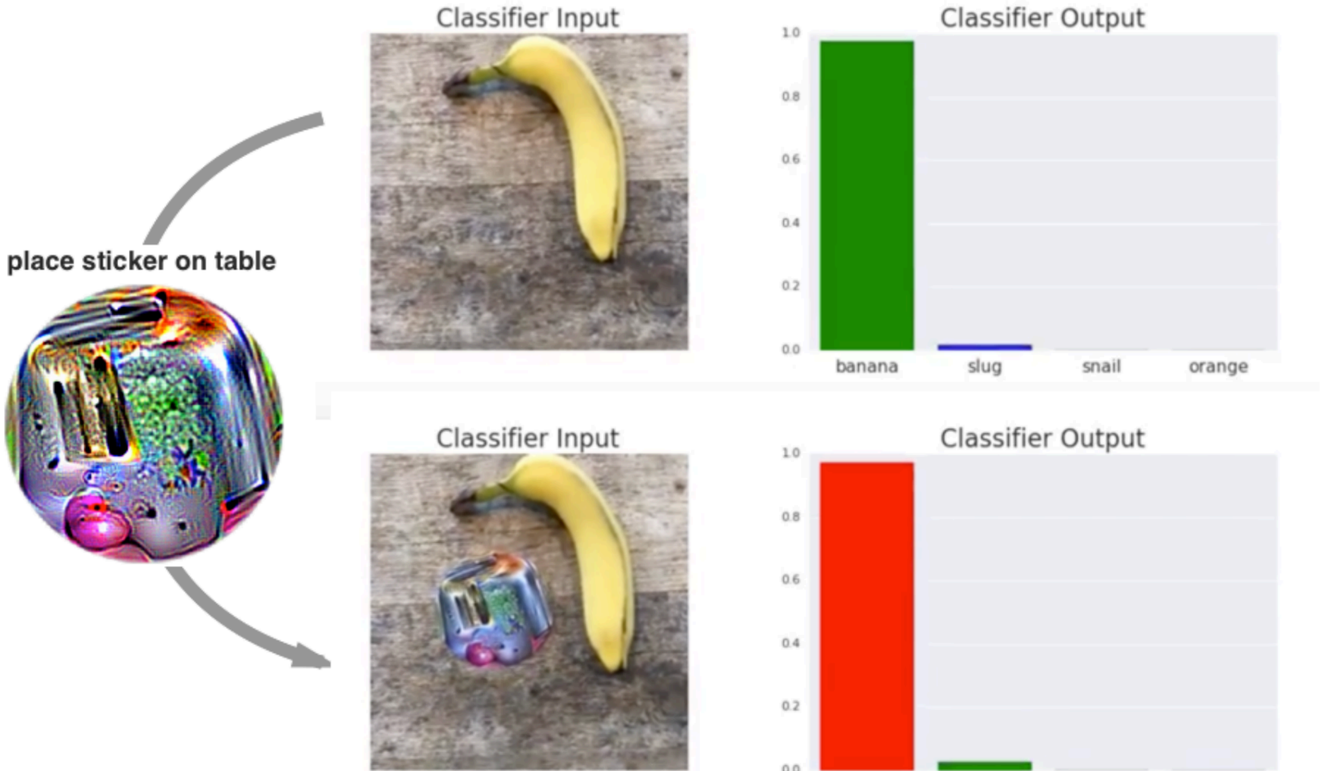
	Available data		
	100%	50%	10%
No augmentation	88.1 (0.32)	85.0 (0.58)	75.1 (0.60)
GAN augmentation	88.4 (0.41)	85.6 (1.33)	76.3 (1.77)
Rotation augmentation	**88.9** (0.51)	**86.0** (0.50)	**76.9** (0.58)
GAN + Rotation augmentation	**89.3** (0.39)	**86.9** (0.36)	**78.4** (0.99)

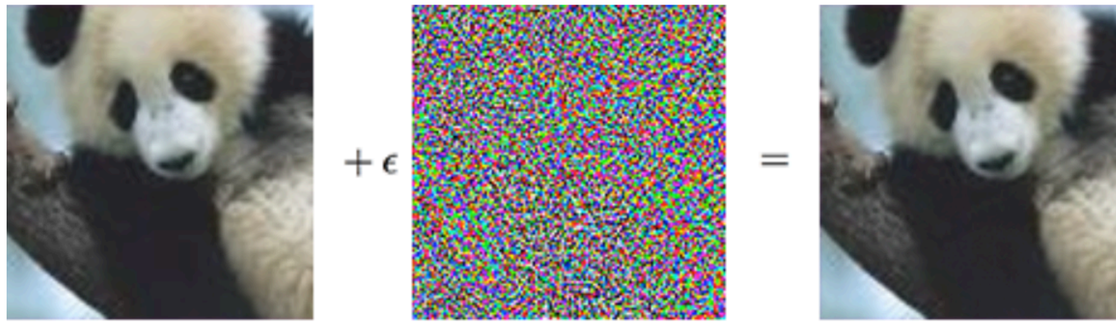


(a) Real CT (b) Synthetic CT (c) Real MRI (d) Synthetic MRI

AL with adversarial examples

- Adversarial examples can “attack” models (https://openai.com/blog/adversarial-example-research/, https://arxiv.org/abs/1712.09665)





“panda”

57.7% confidence

“gibbon”

99.3% confidence

AL with adversarial examples

Same challenge applies for text. E.g. spam-filter fooling, ads, etc...

Question: Can an annotator/process help making a model robust against adversarial attacks? What would a natural query strategy be?

AL with transfer learning

Use a pre-trained model as baseline.

Exercise: Use a model pretrained on ImageNet, and take a seed set of 100 images from CIFAR-10. Use entropy-sampling. How does this compare with no pretraining?

The major question

What would you do ***in practice*** if you were given an large unlabeled dataset?

Example: Classifying the what cuisine a dish is from based on an image.