Deep Learning Models

길 이 만
성균관대학교 소프트웨어대학

Contents

- Overview of Deep Learning
- Convolutional Neural Networks (CNNs)
- Deep Auto Encoders (DAEs)
- Restrictive Boltzmann Machines (RBMs)
- Deep Belief Networks (DBNs)
- Class Probability Output Networks (CPONs)
- Deep Decision Networks (DDNs)
- Concluding Remarks

Deep Learning Overview

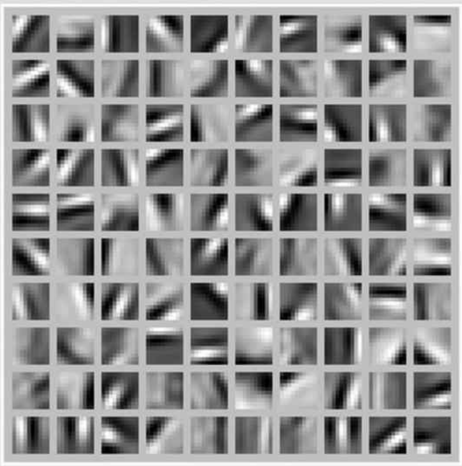
- Train networks with many layers
  (vs. shallow nets with just a couple of layers)

- Multiple layers work to build an improved feature space
  - First layer learns 1st order features (e.g. edges…)
  - 2nd layer learns higher order features (combinations of first layer features, combinations of edges, etc.)
  - In current models layers often learn in an unsupervised mode and discover general features of the input space – serving multiple tasks related to the unsupervised instances (image recognition, etc.)
  - Then final layer features are fed into supervised layer(s)
    - And entire network is often subsequently tuned using supervised training of the entire net, using the initial weights learned in the unsupervised phase
  - Could also do fully supervised versions, etc. (early BP attempts)

3

Deep Learning Tasks

- Usually best when input space is locally structured – spatial or temporal: images, language, etc. vs arbitrary input features
- Images Example: view of vision layer (Basis)



4

Why Deep Learning?

- Biological Plausibility – e.g. Visual Cortex
- Hastad proof - Problems which can be represented with a polynomial number of nodes with k layers, may require an exponential number of nodes with k-1 layers (e.g. parity)
- Highly varying functions can be efficiently represented with deep architectures
  - Less weights/parameters to update than a less efficient shallow representation
- Sub-features created in deep architecture can potentially be shared between multiple tasks

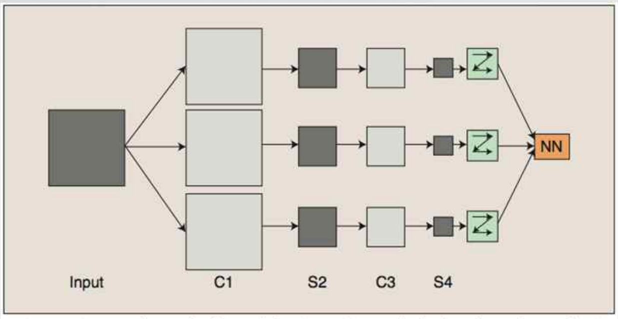
Early Work

- Fukushima (1980) – Neo-Cognitron
- LeCun (1998) – Convolutional Neural Networks
  - Similarities to Neo-Cognitron
- Many layered MLP with backpropagation
  - Tried early but without much success
    - Very slow
    - Diffusion of gradient

Convolutional Neural Networks

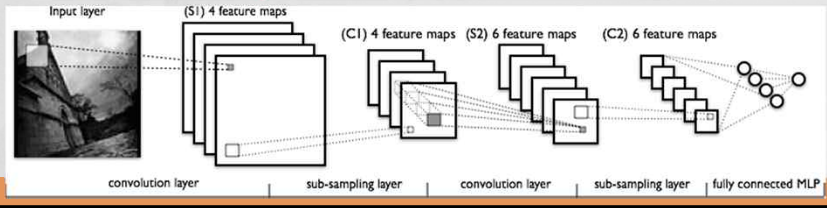
- Each layer combines (merges, smoothes) patches from previous layers
  - Typically tries to compress large data (images) into a smaller set of rob ust features
  - Basic convolution can still create many features
- Pooling
  - This step compresses and smoothes the data
  - Usually takes the average or max value across disjoint patches
- Often convolution filters and pooling are hand crafted – not learned, though tuning can occur
- After this hand-crafted/non-trained/partial-trained convolving the new set of features are used to train a supervised model
- Requires neighborhood regularities in the input space (e.g. images, stationary property)
  - Natural images have the property of being stationary, meaning that th e statistics of one part of the image are the same as any other part

Convolutional Neural Network Examples



C layers are convolutions, S layers pool/sample

FIGURE 2 Conceptual example of convolutional neural network. The input image is convolved with three trainable filters and biases as in Figure 1 to produce three feature maps at the C1 level. Each group of four pixels in the feature maps are added, weighted, combined with a bias, and passed through a sigmoid function to produce the three feature maps at S2. These are again filtered to produce the C3 level. The hierarchy then produces S4 in a manner analogous to S2. Finally these pixel values are rasterized and presented as a single vector input to the "conventional" neural network at the output.



Training Deep Networks

- **Difficulties of supervised training of deep networks**
  - Early layers of MLP do not get trained well
    - Diffusion of Gradient – error attenuates as it propagates to earlier layers
    - Leads to very slow training
    - Need a way for early layers to do effective work
  - Often not enough labeled data available while there may be lots of unlabeled data
    - Can we use unsupervised/semi-supervised approaches to take advantage of the unlabeled data
  - Deep networks tend to have more local minima problems than shal low networks during supervised training

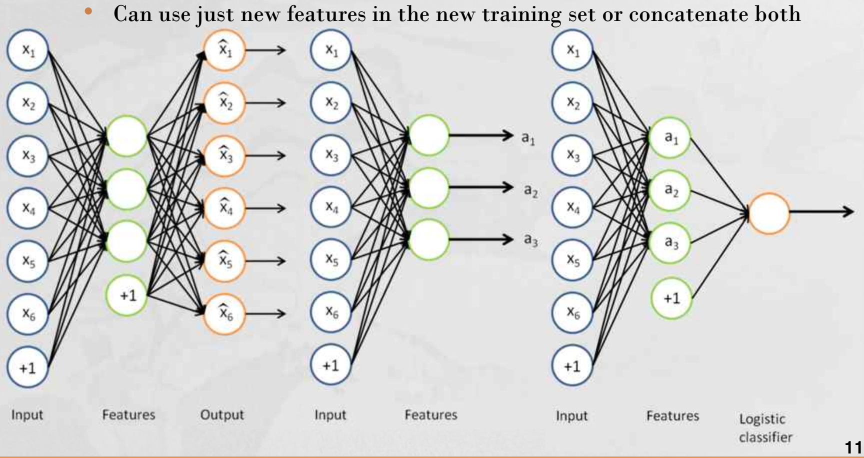
Greedy Layer-Wise Training

- One answer is greedy layer-wise training

1. Train first layer using your data without the labels (unsupervised)
2. Then freeze the first layer parameters and start training the second layer using the output of the first layer as the unsupervised input to the second layer
3. Repeat this for as many layers as desired
   -> This builds our set of robust features
4. Use the outputs of the final layer as inputs to a supervised layer/model and train the last supervised layer(s) (leave early weights frozen)
5. Unfreeze all weights and fine tune the full network by training with a supervised approach, given the pre-processed weight settings

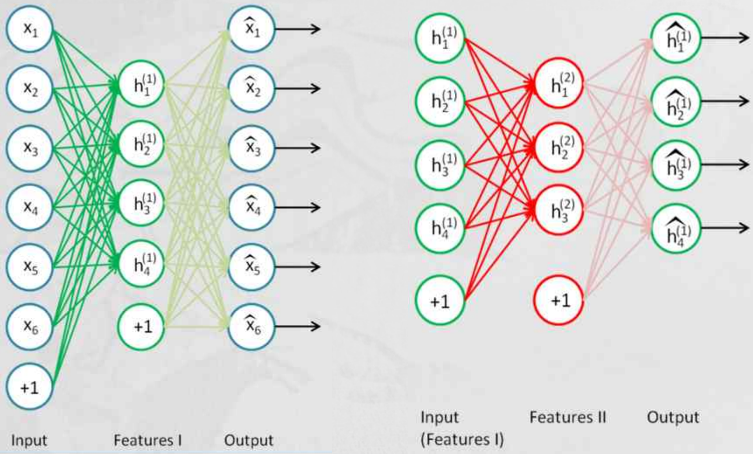
Auto-Encoders

- A type of unsupervised learning which tries to discover generic features of the data
  - Learn identity function by learning important sub-features (not by just passing through data)
  - Compression, etc.
  - Can use just new features in the new training set or concatenate both



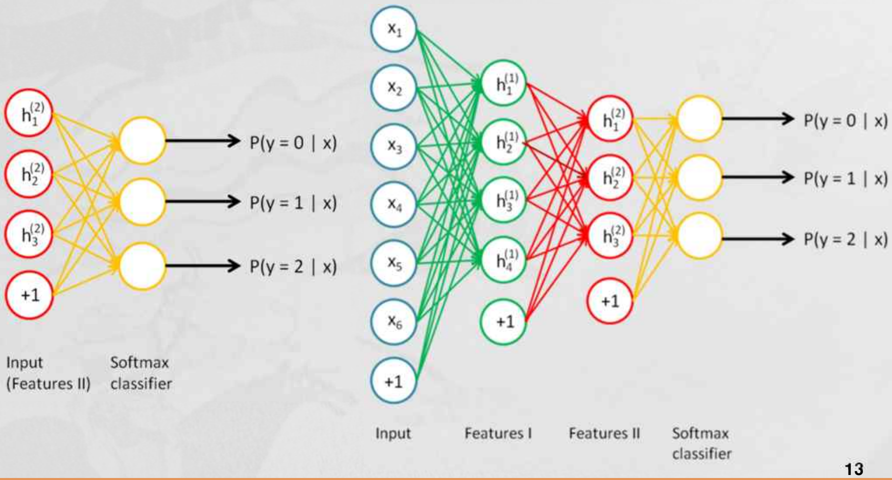
Stacked Auto-Encoders

- Bengio (2007) – After Deep Belief Networks (2006)
- Stack many (sparse) auto-encoders in succession and train them using greed y layer-wise training
- Drop the decode output layer each time



Stacked Auto-Encoders

- Do supervised training on the last layer using final features
- Then do supervised training on the entire network to fine- tune all weights

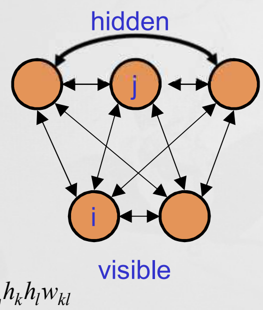


How do we implement a sparse Auto-Encoder?

- Use more hidden nodes in the encoder
- Use regularization techniques which encourage sparseness (e.g. a sig nificant portion of nodes have 0 output for any given input)
  - Penalty in the learning function for non-zero nodes
  - Weight decay
  - etc.
- De-noising Auto-Encoder
  - Stochastically corrupt training instance each time, but still train auto-encoder to decode the uncorrupted instance, forcing it to le arn conditional dependencies within the instance
  - Better empirical results, handles missing values well

Boltzmann Machines

- It is a Undirected graphical model
- The Energy of a joint configuration



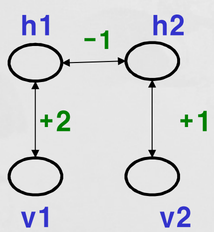
$$-E(\mathbf{v},\mathbf{h}) = \sum_{i \in vis} v_i b_i + \sum_{k \in hid} h_k b_k + \sum_{i<j} v_i v_j w_{ij} + \sum_{i,k} v_i h_k w_{ik} + \sum_{k<l} h_k h_l w_{kl}$$

$$p(\mathbf{v},\mathbf{h}) = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{u},\mathbf{g}} e^{-E(\mathbf{u},\mathbf{g})}} \qquad p(\mathbf{v}) = \frac{\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{u},\mathbf{g}} e^{-E(\mathbf{u},\mathbf{g})}}$$

Boltzmann Machines

$\mathbf{v}$	$\mathbf{h}$	$-E$	e^{-E}	$p(\mathbf{v},\mathbf{h})$	$p(\mathbf{v})$
1 1	1 1	2	7.39	.186	0.466
1 1	1 0	2	7.39	.186	
1 1	0 1	1	2.72	.069	
1 1	0 0	0	1	.025	
1 0	1 1	1	2.72	.069	0.305
1 0	1 0	2	7.39	.186	
1 0	0 1	0	1	.025	
1 0	0 0	0	1	.025	
0 1	1 1	0	1	.025	0.144
0 1	1 0	0	1	.025	
0 1	0 1	1	2.72	.069	
0 1	0 0	0	1	.025	
0 0	1 1	-1	0.37	.009	0.084
0 0	1 0	0	1	.025	
0 0	0 1	0	1	.025	
0 0	0 0	0	1	.025	
			39.70		

An example of how weights define a distribution



Boltzmann Machines

- A very surprising fact

$$\frac{\partial \log p(\mathbf{v})}{\partial w_{ij}} = \langle s_i s_j \rangle_{\mathbf{v}} - \langle s_i s_j \rangle_{model}$$

Derivative of log probability of one training vector, v under the model.

Expected value of product of states at thermal equilibrium when v is clamped on the visible units

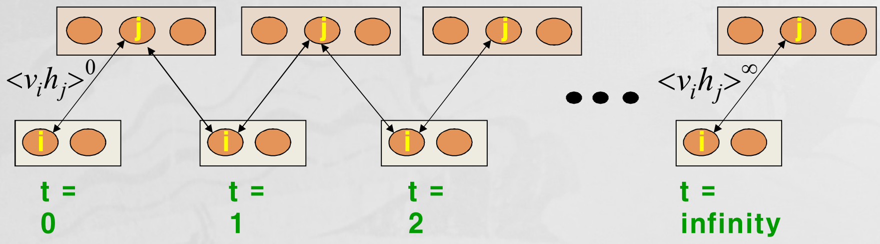
Expected value of product of states at thermal equilibrium with no clamping

$$\Delta w_{ij} \propto \langle s_i s_j \rangle_{data} - \langle s_i s_j \rangle_{model}$$

17

Restricted Boltzmann Machines (RBMs)

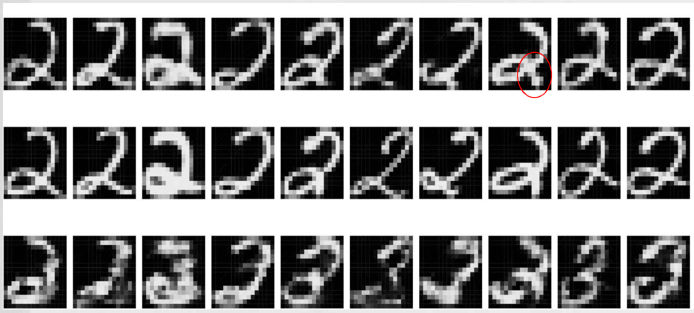
- the Boltzmann machine learning algorithm for an RBM



$$\Delta w_{ij} = \varepsilon\,(<v_i h_j>^0 - <v_i h_j>^\infty)$$

18

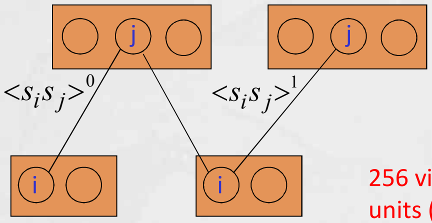
Using an RBM to learn a model of a digit class



Reconstructions by model trained on 2's

Data

Reconstructions by model trained on 3's



100 hidden units (features)

256 visible units (pixels)

Reconstruction of the digit images

Data

Reconstruction from activated binary features

New test images from the digit class that the model was trained on

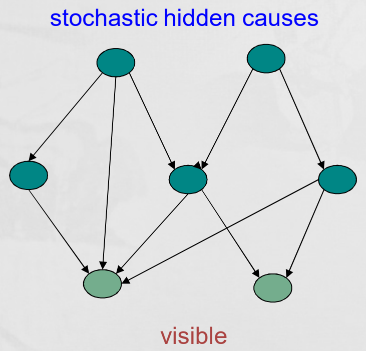
Data

Reconstruction from activated binary features

Images from an unfamiliar digit class (the network tries to see every image as a 2)

Deep Belief Networks (DBNs)

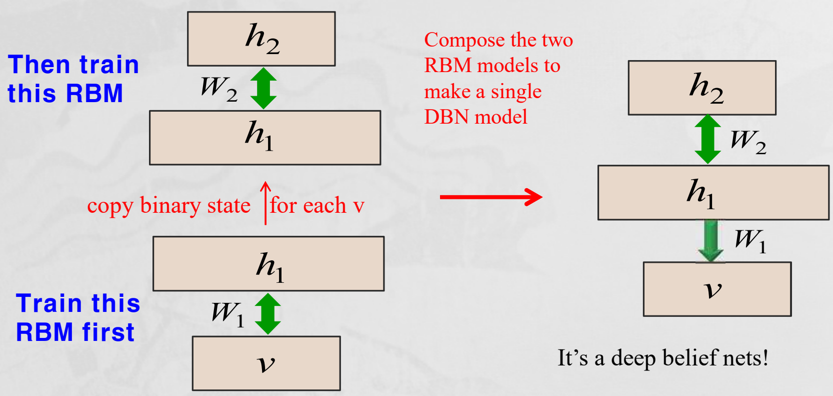
- It is easy to generate sample P(v | h)
- It is hard to infer P(h | v)
  - Explaining away



21

Deep Belief Networks (DBNs)

- Combining two RBMs to make a DBN



It's a deep belief nets!

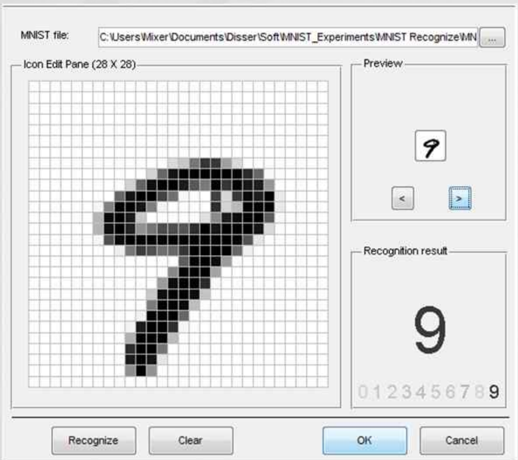
22

Deep Belief Networks (DBNs)

- Discrimination approaches with DBNs (Deep Belief Net)
  - Use outputs of DBNs as inputs to supervised model (i.e. just an unsupervised preprocessor for feature extraction)
  - Train a DBN for each class. For each clamp the unknown x and iterate *m* times. The DBN that ends with the lowest normalized free energy (softmax variation) is the winner.
  - Train just one DBN for all classes, but with an additional visible unit for each class.

Deep Belief Networks (DBNs)

- Character Recognition



Deep Learning

- Issues in deep learning:

  - The storage capacity (or no. of features) of RBMs is too restricted (In the full connection model of BMs, storage capacity is less than 10% of the no. of nodes.

  - The training time of RBMs is usually much larger than the other models of feature extraction.

  - The method of determining the ~~optimal structure~~ (no. of hidden units, no. of layers, etc.) is not known.

Pattern Classification Problems

- Pattern Classification Problems
  - For the given data, find the optimal decision model for data categories.
  - Classification model: linear combination of nonlinear kernels; that is,

$$f(x) = \sum_{i=1}^{m} w_i \phi_i(x)$$

  - Important Questions:

  (1) How do we represent the data distribution?

  (2) What is the optimal decision method for the given data?

Pattern Classification Problems

- The most popular way of implementing pattern classifiers (such as SVM) is using the discriminant function to make a decision of pattern classification.
- In some cases, the discriminant function is used to indicate the degree of confidence for the classification.
- However, the more natural way of representing the confidence for the classification is using the conditional class probability for the decision.

27

Pattern Classification Problems

- There are classifiers that estimate conditional class probabilities such as Parzen window, kernel logistic regression (KLR), Gaussian mixture model (GMM), relevance vector machine (RVM), etc.
- Compared to the classifiers using discriminant functions, the above methods do not show the better classification performances and require higher computational complexity for learning.

28

Pattern Classification Problems

- One method of implementing the better classifier is combining the both types of classifiers.
- From the input to output spaces, use the nonlinear function (such as the linear combination of kernel functions) as the many-to-one mapping.
- Then, the output distribution is modeled by the beta distribution. In principle, this is possible if the joint PDF of input distribution is a continuous function due to the Kolmogorov's theorem of constructing continuous functions.

29

Beta Distribution

However, the Beta function is given by

$$B(\alpha,\beta) = \int_0^1 x^{\alpha-1}(1-x)^{\beta-1}dx = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}.$$

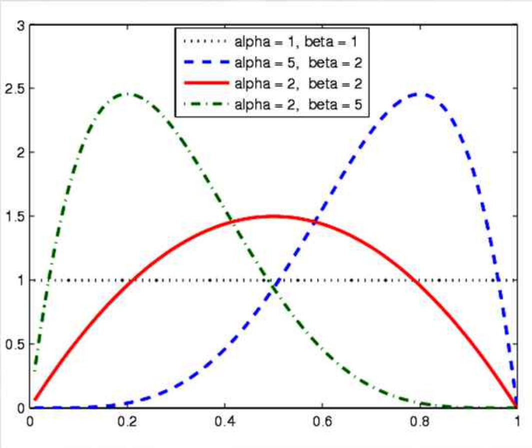
Therefore, after rescaling the PDF of X, we get the Beta PDF described by

$$f_X(x) = \frac{1}{B(\alpha,\beta)}x^{\alpha-1}(1-x)^{\beta-1}, \quad 0 \leq x \leq 1.$$

30

Beta Distribution

- As the parameters α and β vary, the beta distribution takes on many shapes.



Pattern Classification Problems

- **Is there a nonlinear function that converts the input distribution to the Beta distribution?**

  - One simple solution is the cumulative distribution function (CDF) of input distribution. In this case, the output distribution becomes an uniform distribution which is a special case of Beta distributions.

  - Is there another solution except the CDF of input distribution? This is an interesting mathematical problem.

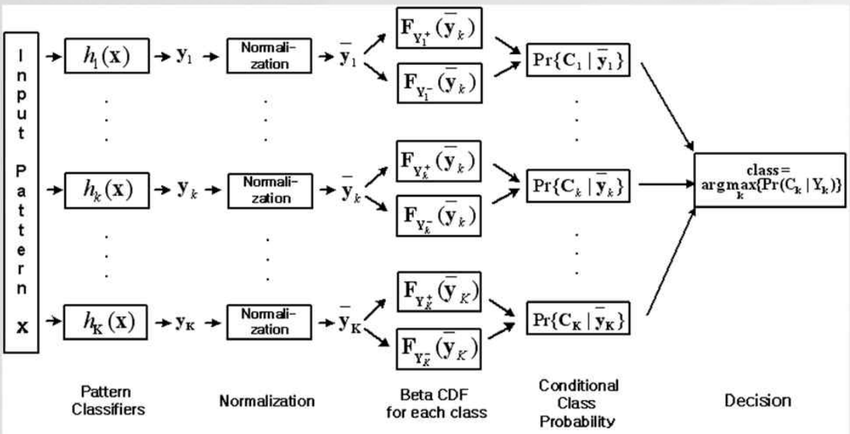
Pattern Classification Problems

- Class Probability Output Network (CPON):

  - First, the classifier using discriminant function such as SVM is trained.
  - The output distribution of classifier is investigated and analyzed using the beta distribution.
  - The parameters of beta distribution as well as the parameters associated with the classifier are adjusted to match with the theoretical data distribution.

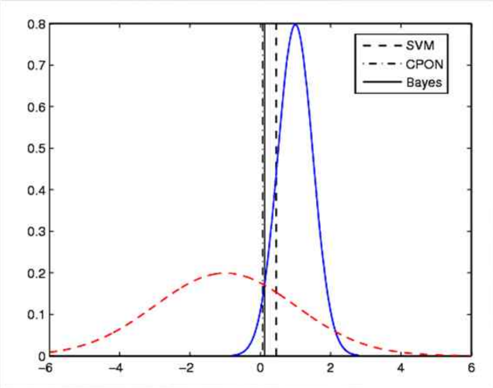
  (distribution matching method)

Class Probability Output Networks



Pattern classification with CPONs

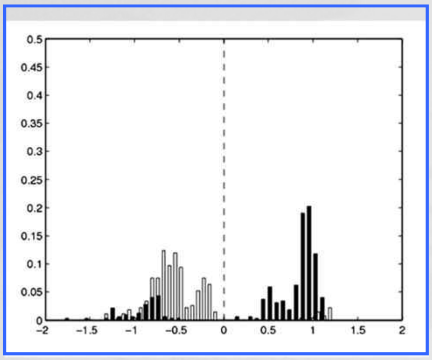
Decision of a class using CPON



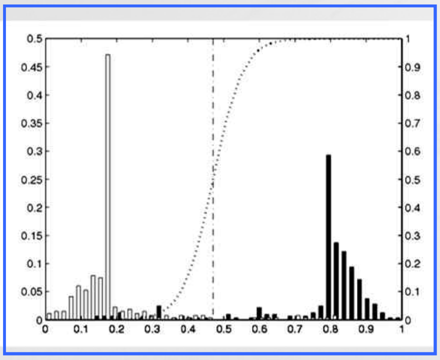
- Train sample: 200 samples :
  100 samples ~ $N(-1,\ 4)$.
  100 samples ~ $N(1,\ 1/4)$.

- Error rate: SVM: 0.1851
  CPON: 0.1639
  Bayes: 0.1635.

Decision of a class using CPON



SVM



CPON

Comparison of output distributions of the SVM and the CPON for the credit approval data set.

Characteristics of CPON

- There is no need to make an assumption on input data distribution.
- The CPON is effective for the unbalanced data sets because the ratio of the number of samples for classes is not important. Rather the sample size itself is important for the accurate estimation of beta parameters.
- The CPON provides better performances than the discriminant function based methods while it offers the degree of confidence for the decision of classification.
- Refs: Park and Kil, IEEE TNN, 2009

  Harvey, Kil, and Han, IEEE TCE, 2010

Active Learning Problems

- **Active Learning Problems**

  - A lot of unlabeled data is plentiful and cheap, for example, documents in the web, speech samples, images and video, etc.

  - But labeling can be expensive or too many data to train.

  - In the opposite case, there are too many data to train the model. Then, what is the good method of training the model? – an important issue in big data problems.

Active Learning Problems

- Two important questions:

  - What is a good measure for the uncertainty of a pattern?

  - Is active learning efficient in terms of sample complexity or generalization bounds?

39

Active Learning Problems

- Active learning algorithms:

  - A^2 algorithm (Balcan, et al., 2006)
  - Disagreement coefficient (Hanneke, 2007)
  - Reduction to supervised (Dasgupta et al., 2007)
  - Importance-weighted approach (Beygelzimer et al., 2009)

40

Active Learning Problems

- Typical procedure of active learning:

Step 1. Start with a pool of unlabeled data

Step 2. Pick a few points at random and get their labels

Step 3. Repeat
Fit a classifier to the labels seen so far
Query the unlabeled data that is closest to the boundary
(or most uncertain, or most likely to decrease overall uncertainty, etc.)

Active Learning Problems

- For the uncertainty of a pattern, the best way may be describing the conditional class probability for the given pattern.

- Furthermore, the confidence interval for the conditional class probability is also an important measure to describe the ambiguity of the decision of the class.

Accuracy of the CPON Output

In the K-S test, a critical value of a test statistic is D_α such that

$$\Pr\{D_n > D_\alpha\} = \alpha \text{ or } \Pr\{\sqrt{n}\,D_n > \sqrt{n}\,D_\alpha\},$$

where α represents the level of significance.

We treat $\sqrt{n}\,D_n = K$ and $\sqrt{n}\,D_\alpha = K_\alpha$.

Then,

$$\Pr\{K > K_\alpha\} = 1 - \frac{\sqrt{2\pi}}{x}\sum_{i=1}^{\infty} e^{-(2i-1)^2\pi^2/(8x^2)} = \alpha.$$

Accuracy of the CPON Output

From the previous equation, the confidence intervals of true CDF values for positive and negative class output of the kth class $F_k^+(\overline{y_k})$ and $F_k^-(\overline{y_k})$ are given as follows: with a probability of $1-\alpha$,

$$F_{Y_k^+}(\overline{y_k}) - D_{\alpha,k}^+ \leq F_k^+(\overline{y_k}) \leq F_{Y_k^+}(\overline{y_k}) + D_{\alpha,k}^+ \quad \text{and}$$

$$(1 - F_{Y_k^-}(\overline{y_k})) - D_{\alpha,k}^- \leq (1 - F_k^-(\overline{y_k})) \leq (1 - F_{Y_k^-}(\overline{y_k})) + D_{\alpha,k}^-.$$

Since the CPON output of the kth class is given by

$$\Pr\{C_k^+ | \overline{y_k}\} = \frac{F_{Y_k^+}(\overline{y_k})}{F_{Y_k^+}(\overline{y_k}) + 1 - F_{Y_k^-}(\overline{y_k})},$$

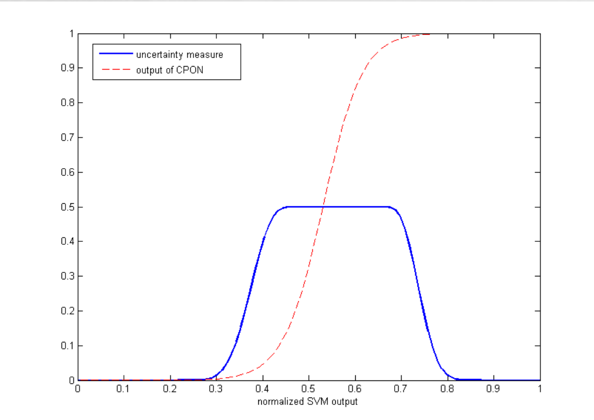
Uncertainty Measure

- In the confidence intervals of the positive and negative classes, the uncertainty measure α is determined when the lower bound of the positive class is same as the upper bound of negative class.
- Then, with a probability of $1-\alpha$,

if the CPON output for positive class > the CPON output for negative class,
p-value of positive class > p-value of negative class,
and vice versa.

Deep Decision Network

An example of uncertainty measure in the CPON output



Active Learning Problems

- The suggested CPON output can be used to identify the possible misclassification for the given pattern since it provides conditional probability estimate.
- The overlapped confidence intervals can be used to determine the uncertainty measure which is needed for the selective sampling.
- Further application of CPON to active learning problems such as the big data problem will be investigated.

Deep Decision Network

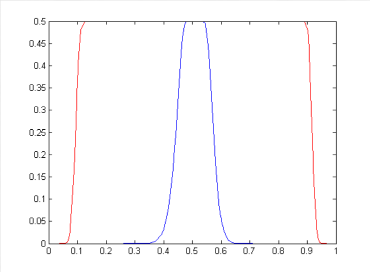
- Alternative Approach: Deep Learning of CPONs

- In the deep network, each hidden unit in a layer provides the condition class probability for the given instance.
- The <u>conditional class probability</u> (or the <u>p-value</u> for testing the hypothesis of classification) is estimated by the CPON.
- The data with high uncertainty (usually greater than 0.01), are propagated in the upper layer and trained by the CPON.
- By propagating conditional class probabilities, the <u>uncertainty of the decision for pattern classification</u> is reduced.

Deep Decision Network

- **Deep Learning using CPONs**

Uncertainty measures for the training of BUPA Liver Disorder Dataset:



Red and blue lines represent the uncertainty measures in the 1st and 2nd layers, respectively.

Deep Decision Network

- **Alternative Approach: Deep Learning of CPONs**

- As stacking layers, the decision for pattern classification becomes near optimal classification (Bayes decision).

- Furthermore, by comparing the uncertainty measure in each layer, the proper number of layers can be determined.

- Ref:
Kim, Yu, Kil, and Lee, Neural Networks, 2015

Concluding Remarks

- The suggested Beta-distribution-based estimation of conditional class probabilities referred to as the CPON was very effective to improve the classification performances of discriminant-function-based classifiers.
- The proposed CPON will also provide effective alternatives to other machine learning models for pattern classification, active learning, deep learning, and other data mining problems including big data problems.

References

- Selected Papers:

- S. Kim, Z. Yu, R. Kil, and M. Lee, "Deep Learning of Support Vector Machines with Class Probability Output Networks," *Neural Networks*, 64:19-28, 2015.
- R. Harvey, R. Kil, and S. Han, "Automatic Media Data Rating Based on Class Probability Output Networks," *IEEE Tr. Consumer Electronics*, 56(4):2296-2302, 2010.
- S. An, R. Kil, and Y. Kim, "Zero-Crossing-Based Speech Segregation and Recognition for Humanoid Robots," *IEEE Tr. Consumer Electronics*, 55(4):2341-2348, 2009.
- W. Park and R. Kil, "Pattern Classification with Class Probability Output Network," *IEEE Tr. Neural Networks*, 20(10):1659-1673, 2009.
- I. Koo and R. Kil, "Model Selection for Regression with Continuous Kernel Functions Using the Modulus of Continuity," *Journal of Machine Learning Research*, 9:2607-2633, 2008.
- I. Koo, N. Lee, and R. Kil, "Parameterized Cross-Validation for Nonlinear Regression Models," *Neurocomputing*, 71:3089-3095, 2008.
- Y. Kim and R. Kil, "Estimation of Inter-aural Time Differences Based on Zero-Crossings in Noisy Multisource Environments," *IEEE Tr. Audio, Speech, and Language Processing*, 15(2):734-743, 2007.
- D. Kim, S. Lee and R. Kil, "Auditory Processing of Speech Signals for Robust Speech Recognition in Real World Noisy Environments," *IEEE Tr. Speech and Audio Processing*, 7(1):55-69, 1999.