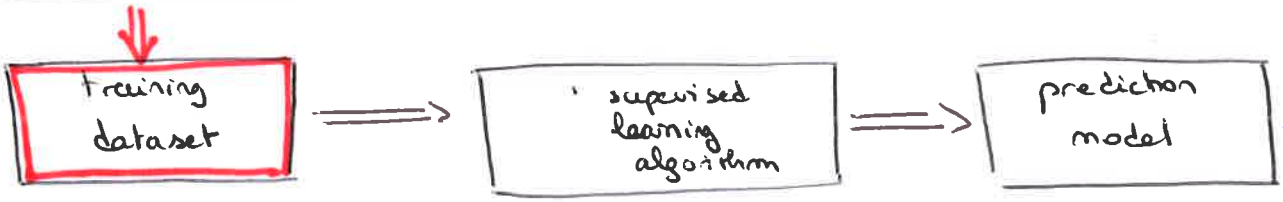
machine learning framework:



set of m input data:
$T = \{(x_\ell^{(1)}, y^{(1)}), \dots (x_\ell^{(m)}, y^{(m)})\}$
with $x_\ell^{(i)} \in \mathbb{R}^m$ for $i = 1 \dots m$
is a feature vector
&
$y^{(i)} \in \{1 \dots C\}$ for $i = 1 \dots m$
is a class label

eg: SVM
ANN
LDA, ...

$h : \mathbb{R}^m \longmapsto \{1 \dots C\}$
$x \longrightarrow y$
discrimination function trying to "mimic" the input-label relationship.

Problem : training dataset is of difficult / expensive to obtain

Goal : take advantage of unlabeled feature data set available :
$\{x_u^{(1)}, \dots x_u^{(k)}\}$, $x_u^{(j)} \in \mathbb{R}^n$ for $j = 1 \dots k$.

eg: see semi-supervised learning [Nigam 2000]
transfer learning [Thrun 1996, Caruana 1997, Ando 2005].

Proposal : improve the training dataset (feature set) using unlabelled data. (multi task learning)

① Use sparse coding to construct a succinct higher-level ("abstract") features representation using unlabeled data (only)

② Apply this representation on labeled dataset and use it for supervised classification.

No assumption on the unlabeled dataset, in particular it doesn't assume that they have the same class labels (eg: task = Rhino vs Elephant, then can use unlabeled img of houses, beaches, horses, pigs ... to try to improve discrimination) nor that feature data follow the same generative distribution.
But it should be relevant eg. same modality, type of data.

Eg: images of Rhinos vs images of Elephants
can images of other scene (eg beach, house, horses, pigs, ...) improve classification of Rhinos vs Elephant?)
⟹ help detect/recognize structures/visual pattern in images (eg edges) / discover salient features they share / correlation between features) ⟹ used for classification task.

Unsupervised feature construction:

unlabelled dataset → high-level/"abstract" representation of features data → "Apply" the new representation → "projected"/optimized training dataset ⇒ supervised learning algorithm

$\{x_u^{(1)}, \dots x_u^{(k)}\}$

$x_u^{(i)} \in \mathbb{R}^m$ for $i = 1..k$

Find a representation basis

$b = \{b_1, \dots b_s\}$

$b_j \in \mathbb{R}^m$ for $j = 1 \dots s$ (can be $\geq m$ overcomplete)

input training dataset

$\{(x_l^{(1)}, y^{(1)}), \dots (x_l^{(m)}, y^{(m)})\}$

$x_l^{(i)} \in \mathbb{R}^m$ for $i = 1..k$

$y^{(i)} \in \{1 \dots c\}$ for $i = 1..k$

$\{(a_l^{(1)}, y^{(1)}), \dots (a_l^{(m)}, y^{(m)})\}$

expres $x_l^{(i)}$ into the new with representation basis. $a_l^{(i)} \in \mathbb{R}^s$, sparse. and

$x_l^{(i)} \simeq a_1^{(i)} b_1 + a_2^{(i)} b_2 + \dots + a_s^{(i)} b_s$

① Construct a representation basis for feature.

Solve:

$$(*) \begin{cases} \min_{b,a} \sum_{i=1}^{k} \| x_u^{(i)} - \sum_{j=1}^{s} a_j^{(i)} b_j \|_2^2 + \lambda \| a^{(i)} \|_1 \\ \text{such that } \| b_j \|_2 \leq 1, \quad \forall j \in 1 \dots s \end{cases}$$

- $\sum_{i=1}^{k} \| x_u^{(i)} - \sum_{j=1}^{s} a_j^{(i)} b_j \|_2^2$: good reconstruction of x_u as a weighted linear combination of b_j - loss function
- $\lambda \| a^{(i)} \|_1$: L_1 - regularisation term to promote sparsity ($\Rightarrow$ most elements $= 0$) (using another norm lead to lower performances).
- $\| b_j \|_2 \leq 1$: constraints

$\sim$ sparse coding algorithm

$\Rightarrow$ come out with a set of | basis vector $b = \{b_1 \dots b_s\}$, $b_j \in \mathbb{R}^m$ for $j = 1 \dots s$ ($s \geq m$ can overcomplete)

| activations $a = \{a^{(1)}, \dots a^{(k)}\}$, $a^{(i)} \in \mathbb{R}^s$ for $i = 1 \dots k$

such that for $i = 1 \dots k$:

$$x_u^{(i)} = a_1^{(i)} b_1 + a_2^{(i)} b_2 + \dots + a_j^{(i)} b_j + a_{j+1}^{(i)} b_{j+1} + \dots + a_s^{(i)} b_s$$

($a_2^{(i)} = 0$, $a_j^{(i)} = 0$, $a_{j+1}^{(i)} = 0$) (see Fig 2&3)

$x_u^{(i)}$ is a sparse weighted combination of b_j - only a few basis elements are used to reconstruct any input

(*) Solved iteratively by alternatingly optimizing over a & b.

(2) Express each training input $x_\ell^{(i)}$, $i = 1 \dots m$, in the new feature representation basis. (3)

Find activations $a(x_\ell^{(i)}) \in \mathbb{R}^s$ for $i = 1 \dots m$

Solve:

$$a(x_\ell^{(i)}) = \underset{a^{(i)}}{\operatorname{argmin}} \left\| x_\ell^{(i)} - \sum_{j=1}^{s} a_j^{(i)} b_j \right\|_2^2 + \beta \|a^{(i)}\|_1$$

note: no constraint, b is fixed, already optimized in step (1).

$$x_\ell^{(i)} = \begin{bmatrix} x_{\ell 1}^{(i)} \\ x_{\ell 2}^{(i)} \\ \vdots \\ x_{\ell m}^{(i)} \end{bmatrix} \Longrightarrow \text{Sparse representation} \quad a(x_\ell^{(i)}) = \begin{bmatrix} a_1^{(i)} \\ a_2^{(i)} \\ \vdots \\ a_s^{(i)} \end{bmatrix} \begin{matrix} = 0 \\ = 0 \\ \\ = 0 \\ = 0 \end{matrix}$$

Comparison to PCA

also used to model "higher-level" structure of the features dataset

Aim: identify a low-dimensional subspace of maximal variation defined by the $k \leq m$ Principal Components $b_1, b_2 \dots b_k$ solution of:

$$\begin{cases} \underset{b,a}{\operatorname{min}} \sum_{i=1}^{m} \| x_u^{(i)} - \sum_{j=1}^{k} a_j^{(i)} b_j \|_2^2 \\ \text{such that } b_1, b_2, \dots b_k \text{ are orthogonal} \end{cases}$$

Advantage: finding $(a_j^{(i)})_{j=1 \dots k}$ corresponding to input $x^{(i)}$ is easy: $a_j(x^{(i)}) = b_j^T x^{(i)}$

Limits:
- linear feature extraction (linear function of the inputs x)
- b_j are assumed to be orthogonal
- $k \leq m$

Experiments

$\{x_u^{(i)}\}_{i=1\dots k}$ $x_u^{(i)} \in \mathbb{R}^m$ —PCA→ $\{b_1^{PCA} \dots b_T^{PCA}\}$ —Sparse coding→ $\{b_1^{sc} \dots b_b^{sc}\}$

$b_i^{PCA} \in \mathbb{R}^m$

$\{a_{PCA}^{(1)} \dots a_{PCA}^{(k)}\}$

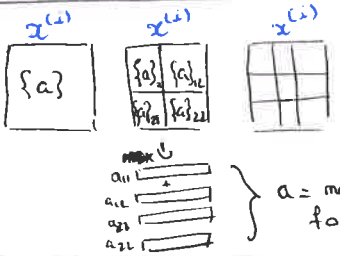
$\{(x_\ell^{(i)}, y^{(i)})\}_{i=1\dots m}$ —"projection"→ $\{(a_{sc}(x_\ell^{(i)}), y^{(i)})\}_{i=1\dots m}$

compare classification learning performances using the following database partitions:

- $\{(a_{sc}(x_\ell^{(i)}), y)\}_{i=1\dots m}$ ie unsupervised feature construction using sparse coding
- $\{(x_\ell^{(i)}, y)\}_{i=1\dots m}$ ie raw features
- $\{(a_{PCA}(x_\ell^{(i)}), y)\}_{i=1\dots m}$ ie PCA features or $\{([x_\ell^{(i)}, a_{PCA}(x_\ell^{(i)})], y)\}_{i=1\dots m}$ (table 3)

→ SVM, GDA

about splitting...



$a^{(i)} = [\max_{i,j}(|a_{i,j=1}|), \max_{i,j}(|a_{i,j=2}|), \dots \max_{i,j}(|a_{i,j=b}|)]$

no statistical significance value.

Learning kernel via Sparse Coding.

fundamental point in ML is defining "similarity" function between 2 examples. One way of formalizing this similarity function is by using a kernel function k: $(x^{(r)}, x^{(t)}) \mapsto k(x^{(r)}, x^{(t)})$ that computes a similarity measure.

eg: SVM can be "kernelized" meaning that the algorithm can be written in an equivalent form where it accesses inputs only via computing the kernel function k between two input examples.

eg: SVM "primal"

$$\begin{cases} \arg\min_{w,\xi} \ \frac{1}{2}\|w\|^2 + \sum \xi \\ \text{s.t.} \quad y_i(w^T x_i + b) \geq 1 - \xi_i \end{cases}$$

$\xi_i \geq 0$

$h(x) = w^T x + b = \sum_{i=1}^{m} \alpha_i y_i \langle x_i, x^T \rangle + b$

SVM "dual"

$$\arg\max_\alpha -\frac{1}{2}\sum_i\sum_j \alpha_i \alpha_j y_i y_j \underbrace{x_i^T x_j}_{\langle x_i, x_j \rangle} + \sum \alpha_i$$

s.t. $0 \leq \alpha_i \leq C$

$\sum \alpha_i y_i = 0$

$\langle x_i, x_j \rangle \nearrow \langle \phi(x_i), \phi(x_j) \rangle \searrow k(x_i, x_j)$

ϕ transformation function

Usually, standard kernels are linear, polynomial or gaussian.

if we change the kernel in the supervised learning algorithm, we can handle other definition of similarity:

$k(x,x) \geq 0$ symetric, ~~semi~~ positive ~~definite~~ bilinear.

Linear kernel: $k(x^{(r)}, x^{(t)}) = x_{(r)}^T x_{(t)}$

polynomial kernel: $k(x^{(r)}, x^{(t)}) = (x_{(r)}^T x_{(t)} + 1)^d$

gaussian kernel (RBF): $k(x^{(r)}, x^{(t)}) = \exp\left(-\frac{\|x^{(r)} - x^{(t)}\|^2}{2\sigma^2}\right)$

→ Build a specific specialized similarity function (kernel) for the learned representation

IDEA: instead of comparing input vectors $x^{(r)}$ and $x^{(t)}$, we can compare the generative process for $x^{(r)}$ and $x^{(t)}$: $P(x^{(r)}|b)$, $P(x^{(t)}, b)$.

ie compare the difference in the way the various generative model parameters b contribute to generating the input vectors $x^{(r)}$ and $x^{(t)}$.

the method compare the gradient of the log input probabilities with respect to the model parameters

Definition

Fisher score: $U_x = \nabla_b \log(P(x|b))$

Fisher kernel: $k(x^{(r)}, x^{(t)}) = U_{x^{(r)}}^T U_{x^{(t)}}$

$P(x|b) = \int_a P(x|b,a)P(a)da \Rightarrow U_x = \nabla_b \log \int_a P(x|b,a)P(a)da.$

point estimate: $\hat{a} = \arg\max_a P(x|b,a)P(a)$ ie find the optimal activations for input x given fixed basis vectors b. (MAP).

...

$U_x = \nabla_b \log(P(x|b,\hat{a})P(\hat{a}))$ ok when derived

Assumption: the standard generative model assumes that the reconstruction error

$\eta = x - Ba$ where $B = [b_1, \ldots b_j \ldots b_s]$

is distributed as a zero-mean Gaussian distribution with covariance $\sigma^2 I$

$P(x|b,a) \propto \exp\left(-\frac{\|\eta\|_2^2}{2\sigma^2}\right)$

Let $\hat{\eta} = x - B\hat{a}$

the method compare the gradient of the log input probabilities with respect to the model parameters

Definition

Fisher score : $U_x = \nabla_b \log(P(x|b))$

Fisher kernel : $k(x^{(r)}, x^{(t)}) = U_{x^{(r)}}^T U_{x^{(t)}}$

$$P(x|b) = \int_a P(x|b,a)P(a)da \Rightarrow U_x = \nabla_b \log \int_a P(x|b,a)P(a)da.$$

point estimate : $\hat{a} = \arg\max_a P(x|b,a)P(a)$ ie find the optimal activations for input x given fixed basis vectors b. (MAP).

...

$$U_x = \nabla_b \log\left(P(x|b,\hat{a})\, P(\hat{a}) \rightarrow\right)$$ ok when derived

Assumption the standard generative model assumes that the reconstruction error

$$\eta = x - Ba \quad \text{where} \quad B = [b_1 \dots b_j \dots b_s]$$

is distributed as a zero-mean Gaussian distribution with covariance $\sigma^2 I$

$$P(x|b,a) \propto \exp\left(-\frac{\|\eta\|_2^2}{2\sigma^2}\right)$$

Let $\hat{\eta} = x - B\hat{a}$

$$\log(P(x|b,\hat{a})) \propto \|\hat{\eta}\|_2^2$$

so $U_x = \nabla_b \log(P(x|b,\hat{a}))\, P(\hat{a}) \propto 2\hat{\eta}\nabla_b\hat{\eta} + 0$ since $\hat{a}$ is not dependant on b!

$\propto \hat{\eta}\hat{a}$ since $\nabla_b\hat{\eta} = \nabla_b(x - B\hat{a})$ constant with respect to a.

Finaly $U_x \propto \hat{\eta}\hat{a} \Rightarrow U_{x^{(r)}}^T U_{x^{(t)}} \propto \hat{a}^{(r)T}\hat{\eta}^{(r)T}\eta^{(t)}a^{(t)} \propto \hat{a}^{(r)T}a^{(t)}\hat{\eta}^{(r)T}\eta^{(t)}$

$$k(x^{(r)}, x^{(t)}) = \underbrace{\left(\hat{a}(x^{(r)})^T a(x^{(t)})\right)}_{\text{inner product of the sparse coding features ie linear kernel of sparse coding features.}} \underbrace{\left(\eta^{(r)T}\eta^{(t)}\right)}_{\text{inner product of the residuals for input examples}}$$

with $\eta = x - \sum_j b_j \hat{a}_j$

optimization

$$\min_{b,a} \sum_i \| x_u^{(i)} - \sum_j a_j^{(i)} b_j \|_2^2 + \beta \| a^{(i)} \|_1 \quad \text{s.t.} \quad \| b_j \|_2^2 \leq 1 .$$

convex in B (while holding A fixed), convex in A (while holding B fixed) but not in both simultaneously.

=> IDEA: iteratively optimize by alternatingly optimizing with respect to B and A (bases) (coefficients) while holding the other fixed.

(1) $\min_b \quad \| X - BA \|^2 \qquad B = [b_1 \dots b_j \dots b_n]$

s.t. $\sum_i b_{ij}^2 \leq 1$

least squares problem with quadratic constraints

↳ eg: gradient descent with iterative projection.

$b' = b - \gamma \nabla F(b)$ where $F(b) = \| X - BA \|$

Here: Solve the DUAL.

(2) $\min_a \quad \| X - BA \|_2^2 + \beta \| A \|_1 \qquad \beta \sum_j |a_j^{(i)}|$

L1-regularized least square problem.

↳ eg: "feature sign search algorithm":

idea: if we know the signs of the $a_j^{(i)}$ at the optimal value, we can replace each of this term $|a_j^{(i)}|$ with $a_j^{(i)}$ / $-a_j^{(i)}$ or 0. => unconstrained quadratic optimization problem (QP)

The algorithm is such that it guess the signs of the coefficient $a_j^{(i)}$ solve a QP ~~and iterate~~ to find optimal $\hat{a}_{new}$ and update the "guess" and iterate.

↓

~~before optimal~~ a_{new}

or signs accordingly