CHEM 313 "Midterm"

General

$\mathcal{X} = \mathbb{R}^D$ Y = prediction

N = #samples T = actual $\sigma(-a) = 1-\sigma(a)$

$\sigma(a) = (1+e^{-a})^{-1}$ $\sigma'(a) = \sigma(a)(1-\sigma(a))$

$\tanh(a) = 2\sigma(a)-1$ $\tanh'(a) = 1-\tanh^2(a)$

$$\underset{\text{posterior}}{p(x|y)} = \frac{\overset{\text{likelihood}}{p(y|x)}\,\overset{\text{prior}}{p(x)}}{p(y)} \quad \Big] \text{evidence}$$

Probability Transform

$Y = f(X)$ $X \sim p_X$ $\int p_X(x)dx = 1$

$X: \Omega \to \mathcal{X}$ $Y: \Omega \to \mathcal{Y}$

$\bar{X}$ = Borel algebra $\bar{Y}$ = Borel algebra

$f: (\mathcal{X}, \bar{X}) \to (\mathcal{Y}, \bar{Y})$

$Y: \Omega \to \mathcal{Y}$ = $Y(\omega) = f(X(\omega))$

$P_Y(B) = P(Y \in B) = P(X \in f^{-1}(B)) = p_X(f^{-1}(B))$

$$p_Y(y) = \int p_X(x)\delta(y-f(x))dx$$

$$= p_X(f^{-1}(y))\left|\frac{d}{dy}(f^{-1}(y))\right| \quad \Big] \text{Monotonic}$$

$$= \sum_{\{x_i | f(x_i) = y\}} \frac{p_X(x_i)}{|f'(x_i)|} \quad \Big] \text{General (IFT)}$$

Dirac

$$\delta(f(x)) = \sum_{\{x_i | f(x_i) = 0\}} \frac{\delta(x-x_i)}{|f'(x_i)|}$$

Normal

$$\mu_{A|B} = \mu_A + \Sigma_{AB}\Sigma_{BB}^{-1}(x_B - \mu_B)$$

$$\Sigma_{A|B} = \Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA}$$

$$\log(p_D(x)) = -\frac{1}{2}\left(\ln(2\pi|\Sigma|) + (x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

Conditional Expectation

$$E[Y|X] = \int y\, p_{Y|X}(x,y)dy$$

$$p_{Y|X}(y,x) = \frac{p_{XY}(x,y)}{p_X(x)}$$

$$\underbrace{p(t|x,w,\beta)}_{\text{likelihood}} = \underbrace{N(t|y(x,w),\beta^{-1})}_{\text{pdf}}$$

Linear

$x_i \in \mathcal{X}$ $y \in \mathbb{R}$

$\phi_i: \mathcal{X} \to \mathbb{R}$

$$y(x,w) = w_0 + \sum_{i=1}^{M-1} w_i\phi_i(x)$$

$$= w_i\phi_i(x)$$

ML Linear

$$\left.\begin{array}{l} t = y + \varepsilon \\ \varepsilon \sim N(0,\beta^{-1}) \end{array}\right] \text{assume}$$

$$p_{t|y}(t,y) = N(t-y,\beta^{-1})$$

$$p_{t|x}(t,x) = N(t|y(x,w),\beta^{-1})$$

$$E[t|x] = \int t\,p(t|x)dt = y(x,w)$$

assuming iid samples $y_n = \phi_i(x_n)w_i$

$$p(\vec{t}|X,w,\beta) = \prod_{n=1}^{N} N(t_n|y_n,\beta^{-1}) \quad \Big] \text{likelihood}$$

log-likelihood

$$\mathcal{L} = \ln(p(t|X,w,\beta)) = \frac{N}{2}\ln(\beta) - \frac{N}{2}\ln(2\pi) - \beta E_D(w)$$

$$E_D(w) = \frac{1}{2}\sum_{n=1}^{N}(t_n - y_n)^2$$

$$\frac{\partial\mathcal{L}}{\partial w_k} = 0 = \sum_n (t_n - w_i\phi_{ni})\frac{\partial}{\partial w_k}(-w_i\phi_{ni})$$

$$= \sum_n (t_n - w_i\phi_{ni})\phi_{nk}$$

$$\phi_{nk}\phi_{ni}w_i = \phi_{nk}t_n$$

$$\phi^T\phi\vec{w} = \phi^T\vec{t}$$

$$\vec{w} = (\phi^T\phi)^{-1}\phi^T\vec{t}$$

Bias: $w_0 = \frac{1}{N}\sum_n\left(t_n - \sum_{j=1}^{M-1} w_j\phi_j(x_n)\right)$

ie offset diff

Regularization

$$E_D(w) + \lambda E_W^q(w) \qquad E_W^q(w) = \frac{1}{2}\sum|w_i|^q$$

"Ridge" $q=2$: $w = (\lambda I + \phi^T\phi)\phi^T\vec{t}$

Lasso:

$$\beta_j = \text{sign}(\beta_j^{\text{Ridge}})\underbrace{\max\left(|\beta_j| - \frac{\lambda}{2}, 0\right)}_{\text{literally kills them on grad}}$$

Multiple Outputs

$y \in \mathbb{R}^K$

$x \in \mathcal{X}$

$y_{nk} = W_{ik}\phi_i(x_n)$

$$\ln(p(t|X,W,\beta)) = \sum_n \ln(N(t_{nk}|W_{ik}\phi_{in},\beta^{-1}I))$$

$$= \frac{NK}{2}\ln\left(\frac{\beta}{2\pi}\right) - \frac{\beta}{2}\sum_{n=1}^{N}\sum_{k=1}^{K}(t_{nk} - y_{nk})^2$$

$$W_{ik} = (\phi_{im}\phi_{jm})^{-1}\phi_{nj}t_{nk} \quad \text{(k free, all indep)}$$

Classification Metrics

Confusion matrix:

A =

True(+)	False(-)
False(+)	True(-)

$$\text{Accuracy} = \frac{A_{ii}}{\sum_{ij} A_{ij}}$$

$$E[L] = \sum_{\underset{\text{classes}}{k}}\sum_{\underset{\text{"decision regions"}}{j}} L_{kj}\int_{R_j} dx\, p(x, C_k)$$

$$p(C_k|x) = \frac{p(x|C_k)p(C_k)}{p(x)}$$

$$p(C_k|x_A, x_B) \propto p(x_A, x_B|C_k)p(C_k)$$

2 features

$$\propto p(x_A|C_k)p(x_B|C_k)p(C_k) \quad \Big] \text{"Naive Bayes" conditional indep. assumption}$$

using to classify $\propto \dfrac{p(C_k|x_A)p(C_k|x_B)}{p(C_k)}$ ← Bayes thm prior

Cross Entropy

$$H(p,q) = \frac{1}{N}\sum_{n=1}^{N}\left(-t_n\ln(y_n) - (1-t_n)\ln(1-y_n)\right)$$

Regression Metrics

$$E[L] = \iint dx\,dt\, L(t, y(x))p(x,t)dxdt$$

usually

$$= \iint dx\,dt\,(t-y)^2 p(x,t)dxdt$$

$$\frac{\delta E}{\delta y} = 2\int(y(x) - t)p(x,t)dt = 0$$

$$y(x)\int p(x,t)dt = \int t\,p(x,t)dt$$

$$y(x)p(x) = \int t\,p(x,t)dt$$

$$\underline{y(x) = E[t|x]} \quad \Big] \text{functionally optimal solution to loss}$$

$$(y(x) - t)^2 = (y(x) - E[t|x])^2 + \underbrace{2(y(x) - E[t|x])(E[t|x] - t)}_{0 \text{ in integral}} + (E[t|x] - t)^2$$

$$E[L] = \int(y(x) - E[t|x])^2 p(x)dx + \quad \Big] \text{bias}$$

$$\int(E[t|x] - t)^2 p(x)dx \quad \Big] \text{variance (irreducible minimum)}$$

"rigid" = few params — high bias + low variance — (high λ)

"flexible" = many params — low bias + high variance — (low λ)

Bayesian Lin Reg

$p(w) = N(w|m_0, S_0)$] prior

$p(w|t) = N(w|m_N, S_N)$ $\quad y_n = w_i \phi_i(x_n)$

$m_{N_i} = S_{N_{ij}}(S^{-1}_{0_{jm}} m_m + \beta \phi_{ni} t_n)$

$S^{-1}_{N_{ij}} = S^{-1}_{0_{ij}} + \beta \phi_{in} \phi_{jn}$ $\quad S_0^{-1} = \alpha I$

$W_{MAP_i} = m_{N_i}$

Ch4 Classificatn

$y(x) = w_i x_i$

2: $y \geq 0 \Rightarrow C_0$
else $\Rightarrow C_1$

K: $\binom{K}{2}$ classifies to distinguish pairs $\quad sign(y) = classif$

$w_i x_i = 0$] hyperplane divides

1 of K Least Squares

$t \in [0,1]^K$ $\quad k = argmax(t)$

$y_k = w_{ik} x_i$

$W^{op}_{ik} = (\phi_{ni} \phi_{nj})^{-1} \phi_{jm} t_{km}$

→ if all t s.t. $\sum t_k = 1$, all predictions will also sum to 1

→ very bad w outliers

Fisher LDA

Intuition: maximize mean separation, minimize in-class variance

ex 2 classes:

$m_{k_i} = \frac{1}{N_k} \sum_{n \in N_k} x_{ni}$

$m_{k_1} - m_{k_2} = w_i(m_{k_1 i} - m_{k_2 i})$ $\quad w_i w_i = 1$ ⇓ orthodirection

$s_k^2 = \sum_{n \in N_k} (w_i x_{in} - m_k)^2$

K=2 $\quad J(w) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$ $\quad S^B_{ij} = (m_{1i} - m_{2i})(m_{1j} - m_{2j})$

max $\quad = \frac{w_i S^B_{ij} w_j}{w_i S^W_{ij} w_j}$ $\quad S^W_{ij} = \sum_{n \in C_1} (x_{ni} - m_{1i})(x_{nj} - m_{1j}) + \sum_{n \in C_2} (x_{ni} - m_{2i})(x_{nj} - m_{2j})$

Optimal: $S^B_{ij} w_j \propto m_{1i} - m_{2i}$

$w_i \propto (S^{W^{-1}})_{ij} (m_{2j} - m_{1j})$ $\quad \Leftrightarrow$ Least squares where for C_1, $t_n = \frac{N}{N_1}$; C_2: $t_n = \frac{-N}{N_2}$

Perceptron

$\theta: \mathbb{R} \to \{-1, 1\}$ (Heaviside)

$y(x) = \theta(w_i \phi_i(x) + b)$] maximally nonlinear

$L = \sum_{n \in misclassified} L_n$

$L_n = -\sum w_i \phi_i(x_n) t_n \geq 0$

Probabilistic Classification

Data → Sigmoid predictor with Gaussians

Assume $p(x|C_k) = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x_i - \mu_{ki}) \Sigma^{-1}_{ij} (x_j - \mu_{kj})\right)$

$p(C_k|x) = \sigma(\underbrace{w_{ik}\phi_i(x) + w_{0k}}_{a_k(x)})$

$w_{ik} = \Sigma^{-1}_{ij} \mu_{jk}$

$w_{0k} = -\frac{1}{2} \mu_{ki} \Sigma^{-1}_{ij} \mu_{kj} + \ln(p(C_k))$

Max-Likelihood

$t_n = 1 \Leftrightarrow C_1$
$t_n = 0 \Leftrightarrow C_2$ $\quad p(C_1) = \pi$

$p(x_n, C_1) = p(C_1) p(x_n|C_1)$
$= \pi N(x_n|\mu_1, \Sigma)$

$p(x_n, C_2) = (1-\pi) N(x_n|\mu_2, \Sigma)$

$p(t|\pi, \mu_1, \mu_2, \Sigma) = \prod_{n=1}^{N} (\pi N(x_n|\mu_1, \Sigma))^{t_n} ((1-\pi) N(x_n|\mu_2, \Sigma))^{1-t_n}$

∴ ML

$\pi = \frac{N_1}{N_1 + N_2}$

$\mu_{ki} = \frac{1}{N_k} \sum t_n^k x_{ni}$

$\Sigma = \frac{1}{N} \sum N_k S_k$

$S_k = \sum_{n \in C_k} (x_{ni} - \mu_{ki})(x_{nj} - \mu_{kj})$

Logistic Regression

$p(C_1|\phi) = y(\phi(x)) = \sigma(w_i \phi_i(x))$

$p(t|w) = \prod_{n=1}^{N} y_n^{t_n} (1 - y_n)^{1-t_n}$

$t_n \in \{0, 1\}$

$E(w) = -\ln p(t|w)$
$= -\sum_n (t_n \ln(y_n) + (1 - t_n) \ln(1 - y_n))$

$\frac{\partial E}{\partial w_j} = \sum_n (y_n - t_n) \phi_j(x_n)$

Procedure

→ Define inductive bias dist
→ Define loss
→ Weights = $-\log(L(\theta|x))$ via some gradient descent

Trees Ch14

$y_{com}(x) = \frac{1}{M}\sum y_m(x)$

mini

Bagging

→ M "bootstrap" datasets are resampled from the original to generate separate datasets/modes which are assiegated via avging

→ mainly to reduce variance (and overfitting) as side effect

Proof

$y_m(x) = h(x) + \varepsilon_m(x)$ underlying

$E[(y_m - h)^2] = E[\varepsilon_m(x)^2]$

$E_{AV} = \frac{1}{M}\sum_m E[\varepsilon_m(x)^2]$] individually

$E_{com} = E\left[\left(\frac{1}{M}\sum_m y_m - h(x)\right)^2\right]$

$= E\left(\frac{1}{M}\sum \varepsilon_m^2\right)$

$\boxed{E_{com} = \frac{1}{M}E_{AV}}$

Assumes

$E[\varepsilon_m(x)] = 0$

$E[\varepsilon_m(x)\varepsilon_n(x)] = S_{nm}\varepsilon_m^2$

Boosting

→ weak learners trained sequentially where if a previous model misclassifies, point weight gets boosted

→ weighted sum @ end

$Y_m = \text{sign}\left(\sum_m a_m y_m(x)\right) \quad t_n \in \{-1, 1\}$

AdaBoost

$$\varepsilon_m = \frac{\sum_n w_n^m I(y_m(x_n) \neq t_n)}{\sum_n w_n^m}$$

$a_m = \ln\left(\frac{1-\varepsilon_m}{\varepsilon_m}\right)$

$w_n^{m+1} = w_n^m \exp(a_m I(y_m(x_n) \neq t_n))$] eg boost missed ones

E_{fun}: exponential

Boosting v. Bagging

trained in sequence	trained indep
improves bias	improves variance

Tree Models (CART

→ binary tree partitions feature space

→ each branch picks the dim & cutoff via exhaustive search greedily

$y_\tau = \frac{1}{N_\tau}\sum_{x_n \in R_\tau} t_n$] optimal prediction

#pts in R_τ

$Q_\tau(T) = \sum_{x_n \in R_\tau} (t_n - y_\tau)^2$

misclassification rate

$C(T) = \sum_{\tau=1}^{|T|} Q_\tau(T) + \lambda|T|$

pruning: ↑ ⇒ too complex/bad

Losses

$Q_\tau(T) = \sum_{k=1}^{K} p_{\tau k}\ln(p_{\tau k})$ Cross entropy

$= \sum p_{\tau k}(1 - p_{\tau k})$ Gini index

$p_{\tau k}$ = proportion of pts in region τ in class k

$p_{\tau k} = 0, 1 \Rightarrow$ good model

$p_{\tau k} = 1/2 \Rightarrow$ bad

→ differentiable unlike missclassification rate

SVD/PCA

$A = USV^\dagger$

$U^\dagger U = I$

$V^\dagger V = I$

$S \geq 0$

$U = eig(AA^\dagger)$ cols = "left sing. vecs"

$V = eig(A^\dagger A)$ cols = "right sing. vecs"

$\|A - A_k\|_F = \sum_{k=K+1}^{r} \sigma_k$

r = rank(A)

$\max_{\|w\|=1} Var(Xw) = w^T S w$

ie top s-vals capture max variance directions (ortho)

$\mathcal{L}(w, \lambda) = w\Sigma w - \lambda(w^Tw - 1)$

$\Sigma w = \lambda w \Leftrightarrow$ PCA

Manifold Techniques

Multidimensional Scaling

→ assumes high dim data actually on a locally smooth low dim manifold

→ Some assume global connectivity

→ eg isomap unrolls

→ eg SNE good for noise/clustering while others bad

$C_{ij} = \delta_{ij} - \frac{1}{N}1_i 1_j$] centering matrix $i \in [N]$

$\hat{X}_{in} = C_{ij}X_{jn}$

$K_{nm} = \hat{X}_{in}\hat{X}_{jm}$

goal: find Z_{zn} s.t. pairwise distances are similar to specified D_{ij} by minimum strain

$\mathcal{L}(Z) = \sum_{nm}\left(K_{nm} - \hat{Z}_{ni}\hat{Z}_{mi}\right)^2$

$\hat{Z}_{in} = C_{ij}Z_{jn}$

(t)SNE

$$p_{j|i} = \frac{\exp\left(\frac{-1}{2\sigma_i^2}|x_i - x_j|^2\right)}{\sum_{k \neq i}\exp\left(\frac{-1}{2\sigma_i^2}|x_i - x_k|^2\right)}$$

prob j would "pick" i as a neighbor in full space

$$q_{j|i} = \frac{\exp(-|z_j - z_i|^2)}{\sum_{k \neq i}\exp(-|z_i - z_j|^2)}$$] low dim z embedding

$\mathcal{L} = \sum_{ij} p_{j|i}\log\frac{p_{j|i}}{q_{j|i}}$

→ pulls far points together rather than push away

swap $\exp(\cdots)$ for $(1 + |x_i - x_j|^2)^{-1}$ in tSNE

Clustering

N_{ij} = # objects in cluster i of class j

$N_i = \sum_j N_{ij} \quad p_{ij} \equiv \frac{N_{ij}}{N_i}$

$p_i \equiv \max_j p_{ij}$

purity $= \sum_i \frac{N_i}{N}p_i$

Hierarchical Agglomerative Clustering

$D_{ij} \geq 0$ input matrix → tree which groups i & j elements

Successively groups most similar then goes up hierarchy until all 1 group

$d_{ij} = \sum_k |x_{ik} - x_{jk}|^2$

or some other metric

K-Means

$K, \mu_k \in \mathbb{R}^p$ minimizes within cluster variance

$z_n^* = \arg\min_k |x_n - \mu_k|^2$

$\mu_{kj} = \frac{1}{N_k}\sum_{n \in C_k} x_{nj}$

$J(M, Z) = \sum_{in}\left(X_{ni} - Z_{nk}M_{ik}\right)^2$] distortion, opt M using alternating minimization

$Z \in \{0,1\}^{NK}$

$X \in \mathbb{R}^{ND}$ data

$M \in \mathbb{R}^{DK}$ k means as columns

Vector Quantization

$J = \frac{1}{N}\sum_n |X_n - \text{decode}(\text{encode}(X_n))|^2$

Quantization

$\text{encode}(x_n) = \arg\min_k |x_n - \mu_k|^2$

$\text{decode}(k) = \mu_k$

Intuition: each vector gets replaced with symbol in codebook

$x_n \in \mathbb{R}^D \to z_n \in [K]$ (snaps them)

$O(NDB) \to O(N\ln K + KDB)$

bits/scalar

nvecs + codebook encoded

Kmeans learns codebook

VQ uses codebook

Kmeans vs GMM

hard	soft (probabilistic assignment)
spherical, equally sized	Gaussian (has own cov)
within cluster variance minimized	maximize Log Likelihood prediction
$\min_{c_k}\sum \|x_n - \mu_{c(n)}\|^2$	$\log p(X) = \sum_n \log\left(\sum_k \pi_k N(x \mid \mu_k, \Sigma_k)\right)$

NNs

ℓ = layer

$z_k^{\ell+1} = h^\ell\left(w_{kj}^\ell z_j^\ell\right) = h^\ell(a_k^\ell)$ nonlinear activation

$z_j^0 = x_j \quad a_k^\ell \quad h^L = \sigma$] classification

$\delta_k^L = \frac{\partial C}{\partial z_k^L}\frac{\partial z_k^L}{\partial a_k^L} = \frac{\partial C}{\partial z_k^L}h'(a_k^L)$

$\frac{\partial a_k^{\ell+1}}{\partial a_j^\ell} = \sum_m w_{km}^\ell \frac{\partial h(a_m^\ell)}{\partial a_j^\ell} = w_{kj}^\ell h'(a_j^\ell)$

$\delta_j^\ell = h'(a_j^\ell)\sum_k w_{kj}^\ell \delta_k^{\ell+1}$

$\delta_k^\ell = \frac{\partial C}{\partial a_k^\ell}$] preactivation error @ layer ℓ

$\frac{\partial C}{\partial w_{kj}^\ell} = \delta_k^{\ell+1} z_j^\ell$

$\frac{\partial C}{\partial a_j^\ell} = \sum \frac{\partial C}{\partial a_k^{\ell+1}}\frac{\partial a_k^{\ell+1}}{\partial a_j^\ell}$] recurse

$p(y|x, w) = \prod_{k=1}^{K} y(x_k w)^{t_k}(1 - y(x_k, w))^{1-t_k}$

K separate bin. cls

$E(w) = -\sum^N\sum^K \left[t_{nk}\ln y_{nk} + (1 - t_{nk})\ln(1 - y_{nk})\right]$ Binary

$E(w) = -\sum^N\sum^K t_{nk}\ln(y_{nk})$

$y_k(x) = \frac{\exp(z_k^L)}{\sum_i \exp(z_i^L)}$

$w^{\tau+1} = w^\tau - \eta\nabla E$

Diagonal Approx

$\frac{\partial^2 E_n}{\partial w_{ji}^2} = \frac{\partial^2 E_n}{\partial a_j^2}z_i^2$ Hessian diagonal

$\frac{\partial^2 E_n}{\partial a_j^2} \sim h'(a_j)^2\sum_k\sum_{k'} w_{kj}^2\frac{\partial^2 E_n}{\partial a_k^2} + h''(a_j)\sum_k w_{kj}\frac{\partial E_n}{\partial a_k}$

$O(W)$ by making diagonal Hessian approx

$E(w) \sim E(\hat{w}) + (w - \hat{w})^T b + \frac{1}{2}(w - \hat{w})^T H(w - \hat{w})$

$H = \frac{\partial^2 E}{\partial w_i \partial w_j}$

$b = \nabla E|_{w=\hat{w}}$

$\nabla E \sim b + H(w - \hat{w})$

$E(w) = E(w^*) + \frac{1}{2}\sum_i \lambda_i \alpha_i^2$

$\sum_i \alpha_i u_i = w - w^*$

also successive differences for H_{L+1}^{-1} from H_L^{-1}

$Hu_i = \lambda_i u_i$

Dropout
→ randomly 0 out weights + rescale remaining on steps so that learns redundancy

Regularization
→ encourages weight sparsity
→ eigenvalue of Hessian $\ll \alpha \rightarrow \sim 0$ $\lambda_i \rightarrow \frac{\lambda_i}{\lambda_i+\alpha_i} \cdot (L_2)$

CNNs
→ translation invariant

$$S_{ij} = (I*K)_{ij} = \sum_{mn} I_{mn} K_{i-m,j-n}$$

img

$$\text{or} = \sum_{mn} I_{mn} K_{i+m,j+n} \quad \text{(cross-correlation)}$$

Convolution $\Longleftrightarrow$ Circulant Matrix mul.

2D Conv $\Longleftrightarrow$ Doubly block circulant matrix

CNNs
→ typically quite sparse elements rather than full img — easier to train
→ param sharing
→ equivariance: successive operations commute
→ f,g equivariant: $f \circ g = g \circ f$
→ convolution & edge detection/scaling/translation commute (equivariant)

Pooling
→ max, avg
→ lazy form of compression
→ roughly invariant to small translations

Invariance
→ presence more useful than location
→ good for "summarizing"
→ ex
→ global GNN readout perm. invariant
→ global pooling

Equivariance
→ able to identify in many orientations
→ encourages parameter sharing, makes respect symmetry, preserve "where/what"
→ ex
→ permutations swap through in GNNs message passing
→ convolution

geometric GNNs have permutational equivariance but also translational/rotational/reflection equivariance

$$h_{ik}^{\ell+1} = \phi_h(h_{ik}^{\ell}, m_{ijk}) \qquad \begin{matrix} k = \text{feature} \\ j,i = \text{node} \end{matrix}$$

$$m_{ik} = \sum_{j \neq i} m_{ijk}$$

$$m_{ijk} = \phi_e(h_{ik}^{\ell}, h_{jk}^{\ell}, a_{ij})$$

Equivariant Version

$$m_{ijk} = \phi_e(h_{ik}^{\ell}, h_{jk}^{\ell}, \|x_i - x_j\|^2, a_{ij})$$

$$x_{ik}^{\ell+1} = x_{ik}^{\ell} + C\sum_{i \neq j} (x_{ik}^{\ell} - x_{jk}^{\ell})\, \phi_x(m_{ijk})$$

Invariant: $f(Tx) = f(x)$

Equivariant: $f(Tx) = Tf(x)$

K fold
→ k sets
→ all used to train + validate in (k-1), 1 sets respectively
→ k=n (leave one out) high variance + costly to run
→ smaller = faster but noisier
→ larger = lower bias