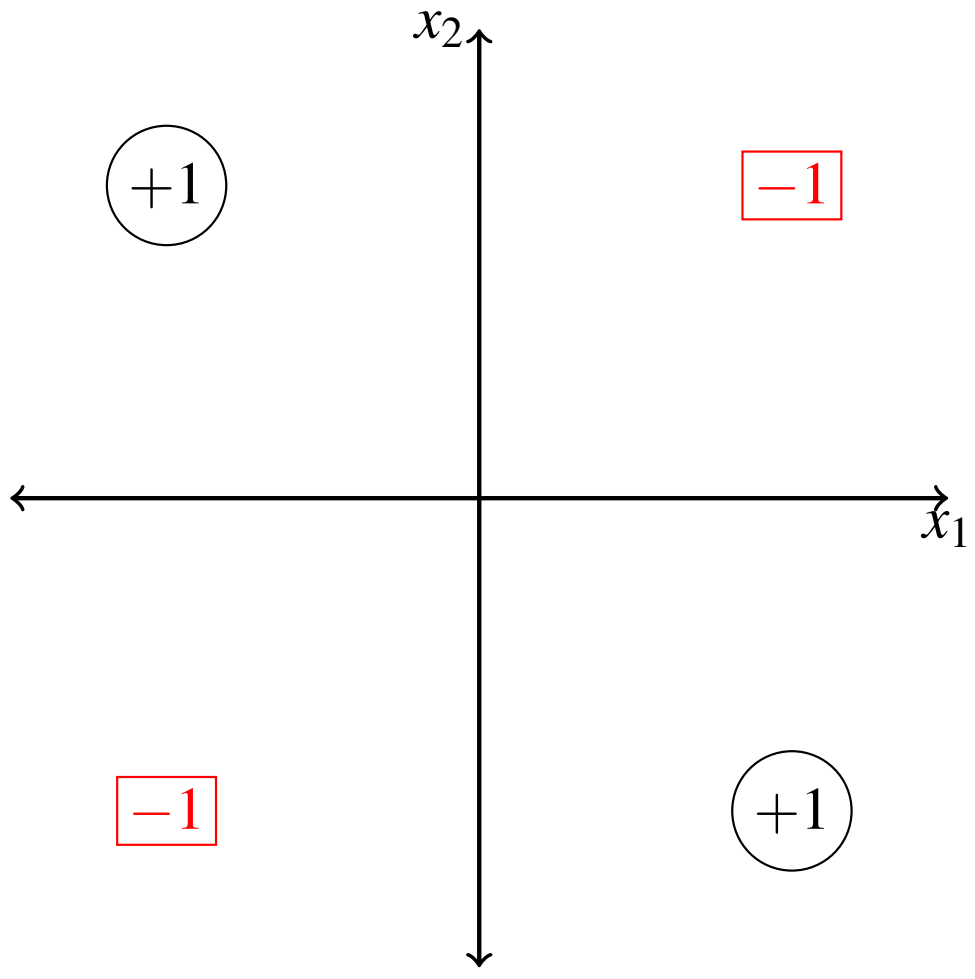
Redes neuronales con funciones de base radial

Diego Milone

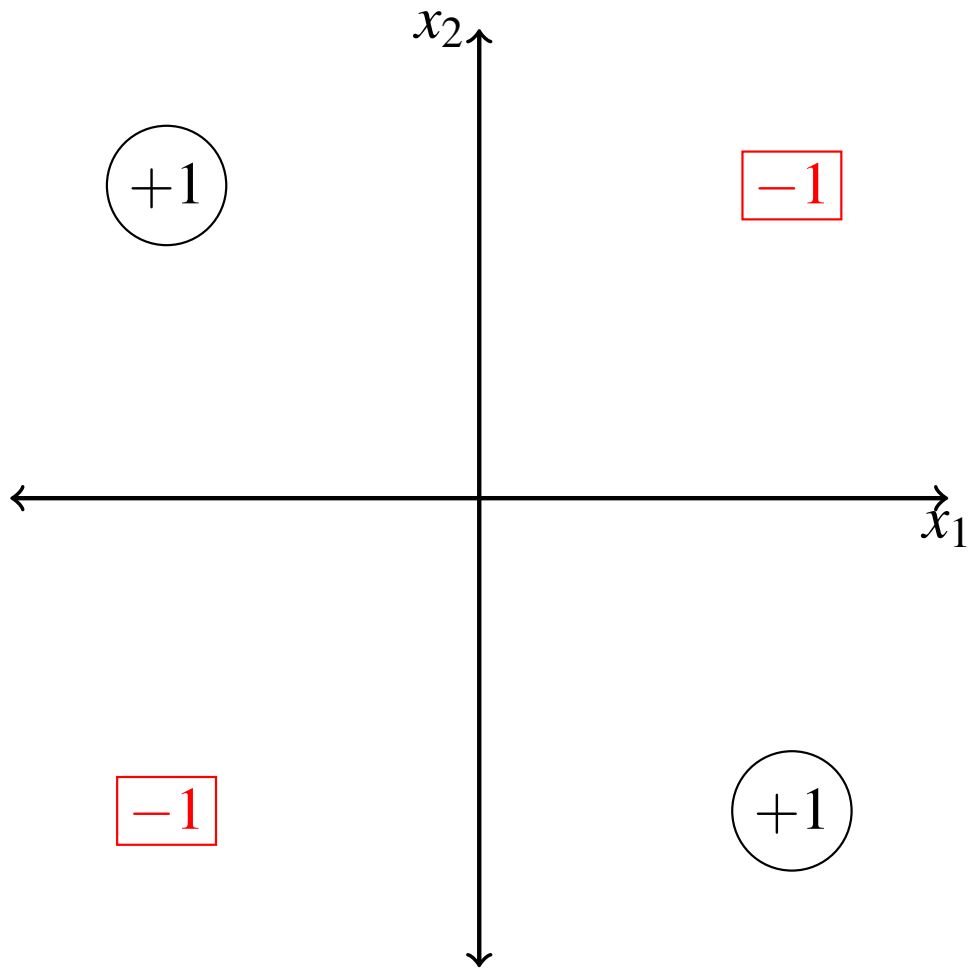
Inteligencia Computacional
Departamento de Informática

FICH-UNL

¿Otra vez el problema XOR?



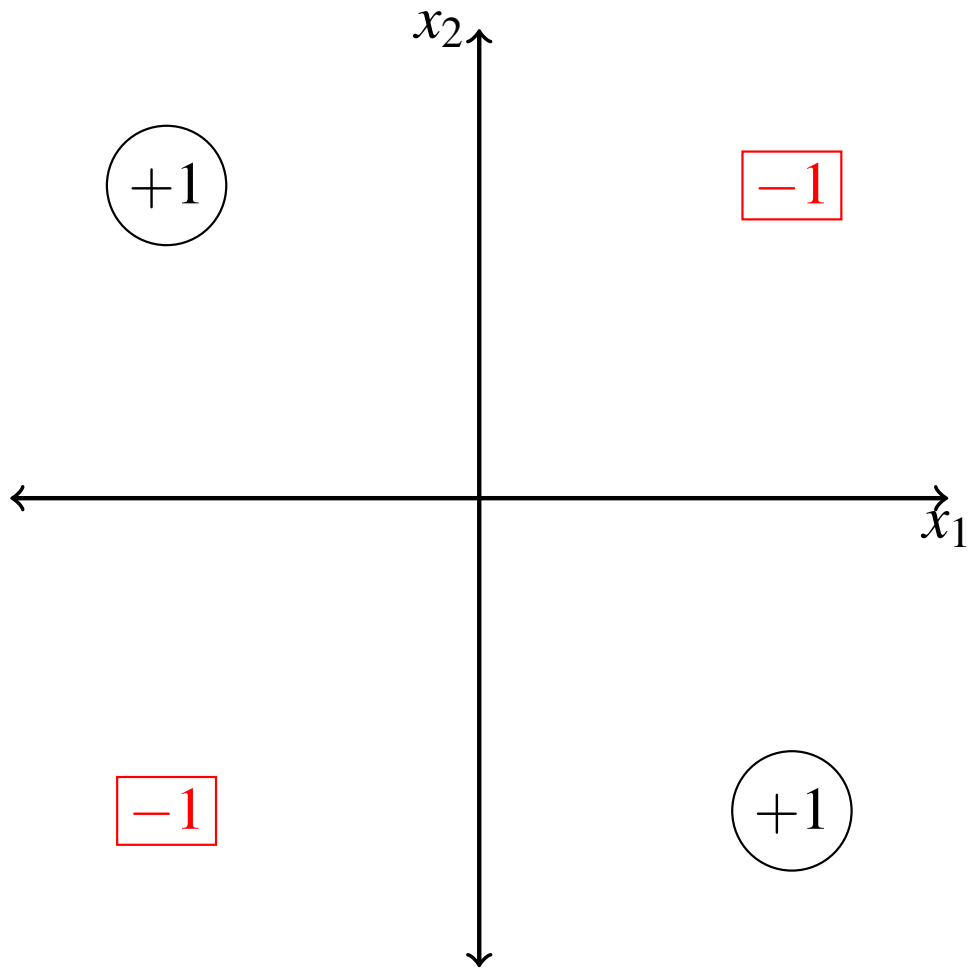
Regiones con un perceptrón multicapa



Funciones sigmoideas

Funciones radiales

Regiones radiales



RBF-NN: introducción

Orígenes de las RBF: aproximación de funciones

$$\phi : \mathbb{R}^N \rightarrow \mathbb{R}$$

$$d = \phi(\mathbf{x})$$

RBF-NN: introducción

Orígenes de las RBF: aproximación de funciones

$$\phi : \mathbb{R}^N \rightarrow \mathbb{R}$$

$$d = \phi(\mathbf{x})$$

Aproximación:

$$h(\mathbf{x}) = \sum_j w_j \phi(\|\mathbf{x} - \boldsymbol{\mu}_j\|)$$

RBF-NN: introducción

Orígenes de las RBF: aproximación de funciones

$$\phi : \mathbb{R}^N \rightarrow \mathbb{R}$$

$$d = \phi(\mathbf{x})$$

Aproximación:

$$h(\mathbf{x}) = \sum_j w_j \phi(\|\mathbf{x} - \boldsymbol{\mu}_j\|)$$

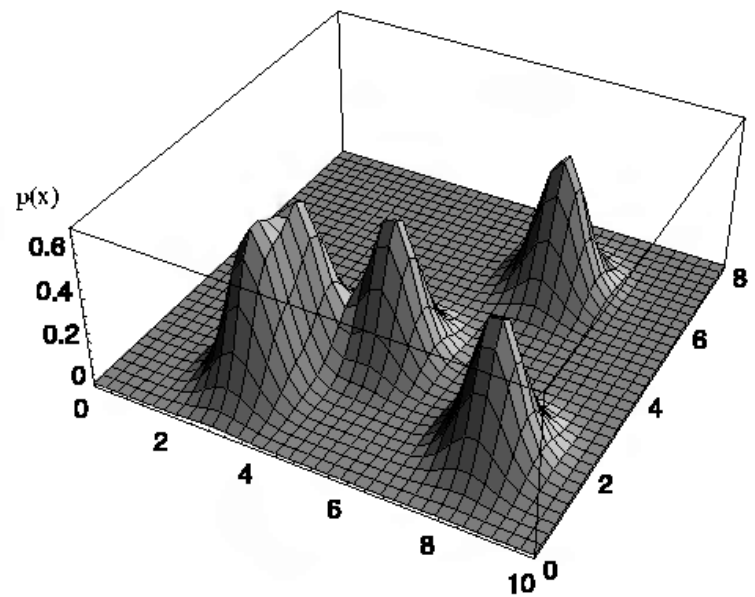
frecuentemente se utiliza como función de base radial:

$$\phi(\kappa) = e^{-\frac{\kappa^2}{2\sigma^2}}$$

RBF-NN: introducción

Aproximación: ejemplo

$$p(\mathbf{x}) = \sum_j w_j \phi(\|\mathbf{x} - \boldsymbol{\mu}_j\|)$$



RBF-NN: generalidades

Arquitectura

RBF-NN: generalidades

Arquitectura

RBF-NN: generalidades

Modelo matemático

$$y_k(\mathbf{x}_\ell) = \sum_{j=1}^{M} w_{kj}\phi_j(\mathbf{x}_\ell)$$

RBF-NN: generalidades

Modelo matemático

$$y_k(\mathbf{x}_\ell) = \sum_{j=1}^{M} w_{kj}\phi_j(\mathbf{x}_\ell)$$

donde:

$$\phi_j(\mathbf{x}_\ell) = e^{-\frac{\|\mathbf{x}_\ell - \boldsymbol{\mu}_j\|^2}{2\sigma_j^2}}$$

RBF-NN: generalidades

Modelo matemático

$$y_k(\mathbf{x}_\ell) = \sum_{j=1}^{M} w_{kj}\phi_j(\mathbf{x}_\ell)$$

donde:

$$\phi_j(\mathbf{x}_\ell) = e^{-\frac{\|\mathbf{x}_\ell - \boldsymbol{\mu}_j\|^2}{2\sigma_j^2}}$$

¿Cuáles son los parámetros a entrenar?

RBF-NN: entrenamiento (parte 1)

Diego Milone

Inteligencia Computacional
Departamento de Informática

FICH-UNL

RBF-NN: entrenamiento

- Método 1:
  - Adaptación no supervisada de las RBF
    - Utilizando el método k-medias
    - Utilizando mapas autoorganizativos
    - Otros...
  - Adaptación supervisada de los w_{kj} (LMS)

RBF-NN: entrenamiento

- Método 1:
  - Adaptación no supervisada de las RBF
    - Utilizando el método k-medias
    - Utilizando mapas autoorganizativos
    - Otros...
  - Adaptación supervisada de los w_{kj} (LMS)

- Método 2:
  - Inicialización por el Método 1
  - Adaptación supervisada de las RBF $\left(\frac{\partial \xi}{\partial \mu_{ji}}, \frac{\partial \xi}{\partial \sigma_j}\right)$

RBF-NN: entrenamiento

- Método 1:
  - Adaptación no supervisada de las RBF
    - Utilizando el método k-medias
    - Utilizando mapas autoorganizativos
    - Otros...
  - Adaptación supervisada de los w_{kj} (LMS)
- Método 2:
  - Inicialización por el Método 1
  - Adaptación supervisada de las RBF $\left(\frac{\partial \xi}{\partial \mu_{ji}}, \frac{\partial \xi}{\partial \sigma_j}\right)$
- En general se adaptan RBF y w_{kj} por separado.

Adaptación de las RBF: *k-medias*

Método NO-supervisado! (uno de los más simples)

Adaptación de las RBF: *k-medias*

Método NO-supervisado! (uno de los más simples)

Objetivos:

- Encontrar k conjuntos C_j de forma que:

Adaptación de las RBF: *k-medias*

Método NO-supervisado! (uno de los más simples)

Objetivos:

- Encontrar k conjuntos C_j de forma que:
  - Cada conjunto C_j sea lo más diferente posible de los demás

Adaptación de las RBF: *k-medias*

Método NO-supervisado! (uno de los más simples)

Objetivos:

- Encontrar k conjuntos C_j de forma que:
  - Cada conjunto C_j sea lo más diferente posible de los demás
  - Los patrones $\mathbf{x}_\ell$ dentro de cada C_j sean lo más parecidos posible entre ellos

Adaptación de las RBF: *k-medias*

Método NO-supervisado! (uno de los más simples)

Objetivos:

- Encontrar k conjuntos C_j de forma que:
  - Cada conjunto C_j sea lo más diferente posible de los demás
  - Los patrones $\mathbf{x}_\ell$ dentro de cada C_j sean lo más parecidos posible entre ellos
- Encontrar el centroide $\boldsymbol{\mu}_j$ de cada conjunto C_j

Adaptación de las RBF: *k-medias*

Método NO-supervisado! (uno de los más simples)

Objetivos:

- Encontrar k conjuntos C_j de forma que:
  - Cada conjunto C_j sea lo más diferente posible de los demás
  - Los patrones $\mathbf{x}_\ell$ dentro de cada C_j sean lo más parecidos posible entre ellos
- Encontrar el centroide $\boldsymbol{\mu}_j$ de cada conjunto C_j

Ecuación de optimización: $\min \left\{ J = \sum_{j=1}^{k} \sum_{\ell \in C_j} \|\mathbf{x}_\ell - \boldsymbol{\mu}_j\|^2 \right\}$

Adaptación de las RBF: *k-medias* por lotes

1. Inicialización: se forman los k conjuntos $C_j(0)$ con patrones $\mathbf{x}_\ell$ elegidos al aleatoriamente.

Adaptación de las RBF: *k-medias* por lotes

1. Inicialización: se forman los k conjuntos $C_j(0)$ con patrones $\mathbf{x}_\ell$ elegidos al aleatoriamente.
2. Se calculan los centroides:

$$\boldsymbol{\mu}_j(n) = \frac{1}{|C_j(n)|} \sum_{\ell \in C_j(n)} \mathbf{x}_\ell$$

Adaptación de las RBF: *k-medias* por lotes

1. Inicialización: se forman los k conjuntos $C_j(0)$ con patrones $\mathbf{x}_\ell$ elegidos al aleatoriamente.
2. Se calculan los centroides:

$$\boldsymbol{\mu}_j(n) = \frac{1}{|C_j(n)|} \sum_{\ell \in C_j(n)} \mathbf{x}_\ell$$

3. Se reasignan los $\mathbf{x}_\ell$ al C_j más cercano:

$$\ell \in C_j(n) \Leftrightarrow \|\mathbf{x}_\ell - \boldsymbol{\mu}_j\|^2 < \|\mathbf{x}_\ell - \boldsymbol{\mu}_i\|^2 \quad \forall i \neq j$$

Adaptación de las RBF: *k-medias* por lotes

1. Inicialización: se forman los k conjuntos $C_j(0)$ con patrones $\mathbf{x}_\ell$ elegidos al aleatoriamente.
2. Se calculan los centroides:

$$\boldsymbol{\mu}_j(n) = \frac{1}{|C_j(n)|} \sum_{\ell \in C_j(n)} \mathbf{x}_\ell$$

3. Se reasignan los $\mathbf{x}_\ell$ al C_j más cercano:

$$\ell \in C_j(n) \Leftrightarrow \|\mathbf{x}_\ell - \boldsymbol{\mu}_j\|^2 < \|\mathbf{x}_\ell - \boldsymbol{\mu}_i\|^2 \quad \forall i \neq j$$

4. Volver a 2 hasta que no se realicen reasignaciones.

Adaptación de las RBF: *k-medias* online

Optimización por método de gradiente:

$$\nabla_{\mu} J = \nabla_{\mu} \left\{ \sum_{j=1}^{k} \sum_{\ell \in C_j} \|\mathbf{x}_{\ell} - \boldsymbol{\mu}_j\|^2 \right\} = 0$$

Adaptación de las RBF: *k-medias* online

Optimización por método de gradiente:

$$\nabla_{\mu} J = \nabla_{\mu} \left\{ \sum_{j=1}^{k} \sum_{\ell \in C_j} \|\mathbf{x}_\ell - \boldsymbol{\mu}_j\|^2 \right\} = 0$$

$$\boldsymbol{\mu}_j(n+1) = \boldsymbol{\mu}_j(n) + \eta(\mathbf{x}_\ell - \boldsymbol{\mu}_j(n))$$

Adaptación de las RBF: *k-medias* online

1. Inicialización: se eligen k patrones aleatoriamente y se usan como centroides iniciales $\boldsymbol{\mu}_j(0) = \mathbf{x}'_\ell$.

Adaptación de las RBF: *k-medias* online

1. Inicialización: se eligen k patrones aleatoriamente y se usan como centroides iniciales $\boldsymbol{\mu}_j(0) = \mathbf{x}'_\ell$.
2. Selección:

$$j* = \arg\min_j \left\{ \|\mathbf{x}_\ell - \boldsymbol{\mu}_j(n)\| \right\}$$

Adaptación de las RBF: *k-medias* online

1. Inicialización: se eligen k patrones aleatoriamente y se usan como centroides iniciales $\boldsymbol{\mu}_j(0) = \mathbf{x}'_\ell$.
2. Selección:

$$j* = \arg\min_j \left\{ \|\mathbf{x}_\ell - \boldsymbol{\mu}_j(n)\| \right\}$$

3. Adaptación:

$$\boldsymbol{\mu}_{j*}(n+1) = \boldsymbol{\mu}_{j*}(n) + \eta(\mathbf{x}_\ell - \boldsymbol{\mu}_{j*}(n))$$

Adaptación de las RBF: *k-medias* online

1. Inicialización: se eligen k patrones aleatoriamente y se usan como centroides iniciales $\boldsymbol{\mu}_j(0) = \mathbf{x}'_\ell$.
2. Selección:

$$j* = \arg\min_j \left\{ \|\mathbf{x}_\ell - \boldsymbol{\mu}_j(n)\| \right\}$$

3. Adaptación:

$$\boldsymbol{\mu}_{j*}(n+1) = \boldsymbol{\mu}_{j*}(n) + \eta(\mathbf{x}_\ell - \boldsymbol{\mu}_{j*}(n))$$

4. Volver a 2 hasta no encontrar mejoras significativas en J.

RBF-NN: entrenamiento (parte 2)

Diego Milone

Inteligencia Computacional
Departamento de Informática

FICH-UNL

Adaptación de los pesos: generalidades

- Al entrenar los pesos, las RBF quedan fijas
- Al estar las RBF fijas se pueden obtener las salidas intermedias para cada patrón de entrada: $\phi(\mathbf{x}_\ell)$
- Con esas salidas intermedias se puede entrenar cada perceptrón simple:

$$\mathbf{y} = \mathbf{W}\phi(\mathbf{x}_\ell)$$

Adaptación de los pesos: generalidades

- Al entrenar los pesos, las RBF quedan fijas
- Al estar las RBF fijas se pueden obtener las salidas intermedias para cada patrón de entrada: $\phi(\mathbf{x}_\ell)$
- Con esas salidas intermedias se puede entrenar cada perceptrón simple:

$$\mathbf{y} = \mathbf{W}\phi(\mathbf{x}_\ell)$$

- Métodos de entrenamiento:
  - pseudo-inversa del vector $\phi(\mathbf{x}_\ell)$
  - gradiente descendiente sobre el error cuadrático instantáneo (LMS)

Adaptación de los pesos: método LMS

$$e_k(n) = y_k(n) - d_k(n)$$

Adaptación de los pesos: método LMS

$$e_k(n) = y_k(n) - d_k(n)$$

$$\xi(n) = \frac{1}{2}\sum_k e_k^2(n) = \frac{1}{2}\sum_k \left(\sum_j w_{kj}(n)\phi_j(n) - d_k(n) \right)^2$$

Adaptación de los pesos: método LMS

$$e_k(n) = y_k(n) - d_k(n)$$

$$\xi(n) = \frac{1}{2}\sum_k e_k^2(n) = \frac{1}{2}\sum_k \left(\sum_j w_{kj}(n)\phi_j(n) - d_k(n) \right)^2$$

$$\frac{\partial \xi(n)}{\partial w_{kj}(n)} = (y_k(n) - d_k(n)) \frac{\partial}{\partial w_{kj}} \left(\sum_j w_{kj}(n)\phi_j(n) - d_k(n) \right)$$

Adaptación de los pesos: método LMS

$$e_k(n) = y_k(n) - d_k(n)$$

$$\xi(n) = \frac{1}{2}\sum_k e_k^2(n) = \frac{1}{2}\sum_k \left(\sum_j w_{kj}(n)\phi_j(n) - d_k(n) \right)^2$$

$$\frac{\partial \xi(n)}{\partial w_{kj}(n)} = (y_k(n) - d_k(n)) \frac{\partial}{\partial w_{kj}} \left(\sum_j w_{kj}(n)\phi_j(n) - d_k(n) \right)$$

$$\frac{\partial \xi(n)}{\partial w_{kj}(n)} = e_k(n)\phi_j(n)$$

Adaptación de los pesos: método LMS

Regla de aprendizaje:

$$w_{kj}(n+1) = w_{kj}(n) - \eta e_k(n)\phi_j(n)$$

$$\boxed{w_{kj}(n+1) = w_{kj}(n) - \eta \left(\sum_i w_{ki}(n)\phi_i(n) - d_k(n) \right) \phi_j(n)}$$

Comparación RBF-NN vs. MLP

RBF-NN

MLP

Comparación RBF-NN vs. MLP

RBF-NN

1 capa oculta

MLP

p capas ocultas

Comparación RBF-NN vs. MLP

RBF-NN

1 capa oculta

distancia a prototipos gaussianos

MLP

p capas ocultas

hiperplanos sigmoideos

Comparación RBF-NN vs. MLP

RBF-NN

1 capa oculta

distancia a prototipos gaussianos

representaciones locales sumadas

MLP

p capas ocultas

hiperplanos sigmoideos

representaciones distribuidas combinadas

Comparación RBF-NN vs. MLP

RBF-NN	MLP
1 capa oculta	p capas ocultas
distancia a prototipos gaussianos	hiperplanos sigmoideos
representaciones locales sumadas	representaciones distribuidas combinadas
convergencia más simple (linealidad)	
entrenamiento más rápido	
arquitectura más simple	

combinación de diferentes paradigmas de aprendizaje

Gaussianas N-dimensionales

Diego Milone

Inteligencia Computacional
Departamento de Informática

FICH-UNL

Anotaciones sobre gaussianas N-dimensionales

Concepto, interpretación gráfica y forma matricial

Forma general $\rightarrow \mathbf{x}, \boldsymbol{\mu}_j \in \mathbb{R}^N, \mathbf{U}_j \in \mathbb{R}^{N \times N}$:

Anotaciones sobre gaussianas N-dimensionales

Concepto, interpretación gráfica y forma matricial

Forma general $\rightarrow \mathbf{x}, \boldsymbol{\mu}_j \in \mathbb{R}^N, \mathbf{U}_j \in \mathbb{R}^{N \times N}$:

$$\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_j, \mathbf{U}_j) = \frac{1}{(2\pi)^{N/2} |\mathbf{U}_j|^{1/2}} \cdot e^{-\frac{1}{2}\left[(\mathbf{x}-\boldsymbol{\mu}_j)^T \mathbf{U}_j^{-1} (\mathbf{x}-\boldsymbol{\mu}_j)\right]}$$

Anotaciones sobre gaussianas N-dimensionales

- Caso simplificado 3 $\rightarrow \mathbf{U}_j = \mathbf{I}$:

$$\mathcal{N}'(\mathbf{x}, \boldsymbol{\mu}_j) = e^{-\frac{1}{2}\sum_{k=1}^{N}\left(x_k - \mu_{jk}\right)^2}$$

Anotaciones sobre gaussianas N-dimensionales

- Caso simplificado 3 $\rightarrow \mathbf{U}_j = \mathbf{I}$:

$$\mathcal{N}'(\mathbf{x}, \boldsymbol{\mu}_j) = e^{-\frac{1}{2}\sum_{k=1}^{N}\left(x_k - \mu_{jk}\right)^2}$$

- Caso simplificado 2 $\rightarrow \mathbf{U}_j \in \mathbb{R}^{N\times N}$, diagonal igual:

$$\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_j, \mathbf{U}_j) = \frac{1}{(2\pi)^{N/2}\sqrt{N}\sigma} \cdot e^{-\frac{1}{2\sigma^2}\sum_{k=1}^{N}\left(x_k - \mu_{jk}\right)^2}$$

Anotaciones sobre gaussianas N-dimensionales

- Caso simplificado 1 $\rightarrow \mathbf{U}_j \in \mathbb{R}^{N \times N}$, diagonal general:

$$\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_j, \mathbf{U}_j) = \frac{1}{(2\pi)^{N/2}\sqrt{\sum_{k=1}^{N} \sigma_{jk}^2}} \cdot e^{-\frac{1}{2}\sum_{k=1}^{N} \frac{(x_k - \mu_{jk})^2}{\sigma_{jk}^2}}$$

Anotaciones sobre gaussianas N-dimensionales

- Caso simplificado 1 $\rightarrow \mathbf{U}_j \in \mathbb{R}^{N \times N}$, diagonal general:

$$\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_j, \mathbf{U}_j) = \frac{1}{(2\pi)^{N/2}\sqrt{\sum_{k=1}^{N} \sigma_{jk}^2}} \cdot e^{-\frac{1}{2}\sum_{k=1}^{N} \frac{(x_k - \mu_{jk})^2}{\sigma_{jk}^2}}$$

- Forma general $\rightarrow \mathbf{x}, \boldsymbol{\mu}_j \in \mathbb{R}^N, \mathbf{U}_j \in \mathbb{R}^{N \times N}$:

$$\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_j, \mathbf{U}_j) = \frac{1}{(2\pi)^{N/2}|\mathbf{U}_j|^{1/2}} \cdot e^{-\frac{1}{2}\left[(\mathbf{x} - \boldsymbol{\mu}_j)^T \mathbf{U}_j^{-1} (\mathbf{x} - \boldsymbol{\mu}_j)\right]}$$