Universidad de Córdoba

Escuela Politécnica Superior de Córdoba
Grado en Ingeniería informática

Práctica 3. Red neuronal de funciones de base radial

Introducción a los modelos computacionales
Cuarto curso de Computación

Por: Francisco Javier Pérez Castillo
Correo: i72pecaf@uco.es
DNI: 32734629V

Índice de la memoria

1. Entrenamiento de la red RBF

En una red neuronal de funciones de base radial presentan neuronas con funciones, como su nombre indica, de base radial. Son funciones locales que en capa oculta realizan transformaciones locales no lineales, poseen centro, el cual es el punto simétrico de la función y una anchura o radio, que mide la variación según nos alejamos del centro. En estas funciones, cuanto más nos acercamos al centro de la misma, más nos aproximamos al valor máximo que pueden tomar.

No existe una única forma de entrenar una red de este estilo. Las funciones radiales que podemos aplicar para este modelo son posibles de derivar, por lo que se presenta la posibilidad de aplicar el algoritmo de retropropagación. Pero también hay otra alternativa, un entrenamiento híbrido.

La manera alternativa es un aprendizaje de forma híbrida, esto es, aplicando una fase no supervisada y una fase supervisada.

El aprendizaje de esta red consiste en determinar los parámetros de la misma, los cuales son los centros y radios de funciones radiales, que se encuentran en las neuronas de capa oculta, y la obtención de los pesos de la capa de salida, que usan los resultados de las funciones de base radial.

Para la obtención de los parámetros de las funciones radiales aplicaremos un aprendizaje no supervisado, aplicando un clustering. Se buscará clasificar todo el espacio de entrada (patrones) en diferentes clases, aplicando para ello el algoritmo de K-medias. Con esto, obtendremos un conjunto de puntos que son los centros de cada cluster y que, para nuestras funciones, serán los centros de los radios. Para nuestra práctica, consideramos que el número de clusters será el mismo que el número de neuronas RBF.

Con esto obtendremos los centros, pero no los radios, para ello tomaremos como radio la distancia media al resto de centros.

En el caso de la fase supervisada, tendremos dos formas de obtener los pesos de los enlaces a las salidas dependiendo sí nos encontramos con un problema de clasificación o de regresión.

Si nos encontramos con un problema de clasificación deberemos de aplicar una regresión logística para las salidas de las neuronas de capa oculta, con el objetivo de obtener los coeficientes beta de la función softmax, los cuales posibilitan maximizar la entropía cruzada

En la regresión, buscaremos calcular la inversa de una matriz no cuadrada, para ello, utilizaremos la matriz pseudoinversa de Moore Penrose.

2. Bases de datos y parámetros usados

2.1 Bases de datos usadas

En la realización de los experimentos de esta práctica han tomado una serie de bases de datos divididas en problemas de clasificación, para *noMNIST* y para *divorce*, y en problemas de regresión, para el resto. Ahora se explicarán brevemente las bases de datos una por una.

· *Función seno*: Base de datos compuesta por 120 patrones de entrenamiento y 41 de test. Se le ha agregado ruido aleatorio. Posee una variable de entrada y una única variable de salida.

· *Quake*: Posee 1633 patrones de entrenamiento y 546 de test. El objetivo de la misma es medir la intensidad (fuerza) que posee un terremoto en la escala Richter. La profundidad focal, la latitud donde se produce el fenómeno y la longitud son sus variables de entrada, poseyendo una única salida.

· *Parkisons*: Alberga 4406 patrones de entrenamiento y 1469 para la fase de test. Para sus entradas encontramos información clínica de pacientes que poseen Parkinson y otras medidas biométricas como la voz. Para la salidas tenemos el valor motor y el total de UPDRS, teniendo un total de 19 variables de entrada y dos de salida.

· *Divorce*: La base de datos posee 127 patrones para el entrenamiento y 43 para el test. Como variables de entrada tenemos variables numéricas que representan valores en escala Likert (entre 0 y 4) que son las respuestas a preguntas de encuestas que tienen el objetivo de conocer si se va a producir un divorcio o no. En total son 54 variables y una salida.

· *noMNIST*: Se encuentran almacenados 900 patrones de entrenamiento y 300 de test. Recopila las letras de la A a la F escritas con diferentes tipologías ajustadas en una rejilla cuadrada de 28 x 28 píxeles en una escala de grises en el intervalo [1,0 ; +1,0].

2.2 Parámetros usados

Esta práctica ha sido desarrollada utilizando python para la creación de un programa que reproduce una red RBF. Este programa posee una serie de parámetros que serán descritas a continuación.

→ -t , --train_file : Parámetro obligatorio que recibe la ruta del fichero de la base de datos para el entrenamiento

→ -T , --test_file : Parámetro que recibe la ruta del fichero de la base de datos para entrenamiento. Si no se especifica uno, se tomará el de test.

→ -c , --classification : Parámetro booleana que indica con valor *True* que el problema será de clasificación. Si no se introduce valor, el problema será de regresión.

→ -r , --ratio_rbf : Número de neuronas que se usarán, se calcula en base al número de patrones usados. Se ha de pasar un valor en tanto por uno que indicará la proporción de estas.

→ -l , --l2 : Indica con *True* o *False* (Booleano) si se utilizará la regularización L2 en lugar de L1. Si no especificamos nada, se tomará L1.

→ -e , --eta : Indica el valor del parámetro eta (η). Por defecto, se usa η = exp(-2)

→ -o , --outputs : Indica el número de columnas de salida que tiene la base de datos y que están al final del fichero. Por defecto se tomará como que hay una única salida.

→ -h , --help : Se mostrará la ayuda del programa.

3. Resultados obtenidos y análisis

En esta práctica probaremos la red aplicando cuatro experimentos distintos y obteniendo resultados para todos los patrones anteriormente expuestos.

3.1 Primer experimento

El objetivo de esta prueba es encontrar los mejores resultados modificando la arquitectura de la misma, osea sé, modificando el número de neuronas en capa oculta. Dado que no puede variar el número de capas ya que nos encontramos en una red de tipo RBF, este será el único parámetro que permita modificar la arquitectura de la misma. Los mejores resultados indicarán la mejor arquitectura para cada base de datos. Se tomará para todos las bases de datos $\eta = 10^{-5}$ y para los problemas de clasificación, L1 como regularización.

Los resultados son los siguientes:

Función Seno

Ratio RBF	***Training MSE***	***Test MSE***
5%	0.013817 +- 0.000189	0.022182 +- 0.000245
15%	0.011216 +- 0.000068	0.011216 +- 0.000068
25%	0.011216 +- 0.000068	0.011216 +- 0.000068
50%	0.010357 +- 0.000002	2.161204 +- 0.544252

Quake

Ratio RBF	***Training MSE***	***Test MSE***
5%	0.028365 +- 0.000063	0.028475 +- 0.000223
15%	0.025321 +- 0.000120	0.040689 +- 0.001947
25%	0.022196 +- 0.000079	0.935580 +- 0.290723
50%	0.018456 +- 0.000082	158.875154 +- 60.571705

Parkinsons

Ratio RBF	***Training MSE***	***Test MSE***
5%	0.001670 +- 0.000067	0.002037 +- 0.000055
15%	0.000749 +- 0.000010	0.001289 +- 0.000081
25%	0.000212 +- 0.000017	0.001064 +- 0.000043
50%	0.000333 +- 0.000002	0.00327 +- 0.000132

Divorce

Ratio RBF	***Training MSE***	***Test MSE***	***CCR Train***	***CCR Test***
5%	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
15%	0.000001 +- 0.000000	0.000008 +- 0.000008	100.00% +- 0.00%	100.00% +- 0.00%
25%	0.000001 +- 0.000001	0.004103 +- 0.005523	100.00% +- 0.00%	99.53% +- 0.93%
50%	0.000001 +- 0.000001	0.004905 +- 0.004738	100.00% +- 0.00%	99.53% +- 0.93%

noMNIST

Ratio RBF	***Training MSE***	***Test MSE***	***CCR Train***	***CCR Test***
5%	0.029815 +- 0.001447	0.029202 +- 0.001339	88.09% +- 0.54%	88.67% +- 0.87%
15%	0.000377 +- 0.000211	0.043791 +- 0.003821	100.00% +- 0.00%	86.47% +- 1.39%
25%	0.000033 +- 0.000004	0.039877 +- 0.002755	100.00% +- 0.00%	87.00% +- 0.82%
50%	0.000038 +- 0.000002	0.035192 +- 0.001981	100.00% +- 0.00%	87.93% +- 0.49%

Se puede ver que el error medio en los problemas de regresión presentan problemas cuando aumenta el número de neuronas en capa oculta.

Observando los datos referidos a los problemas de clasificación podemos ver que un menor ratio mejora los resultados del MSE y del CCR, pero en el caso de noMNIST se puede observar que al aumentar el número de neuronas vemos que mejoran los resultados.

Estos problemas pueden deberse a la calidad de la información utilizada, que influye enormemente en la información tratada. Una posible interpretación es que al encontrarse muchos patrones con muchos centroides, muchos de estos se agrupan incorrectamente, tampoco el casó contrario es óptimo, ya que resultaría en muchos patrones distintos pueden ir a un mismo centroide.

Nota: Encuentro una anomalía en los resultados de noMNIST entre el 25% y el 50% del ratio, en un principio entiendo que deberían de disminuir el error tanto en training como en test, sumado a que el CCR del test aumenta brevemente con respecto al anterior.

3.2 Segundo experimento

En este experimento probaremos a cambiar los valores de eta y probaremos los dos tipos de regularización, L1 y L2, para los problemas de clasificación. Tomaremos, como la mejor arquitectura, un ratio de 5%

Divorce para L1

η	***Training MSE***	***Test MSE***	***CCR Train***	***CCR Test***
1	0.019916 +- 0.000734	0.021780 +- 0.000447	97.64% +- 0.00%	97.67% +- 0.00%
10^{-1}	0.005612 +- 0.001197	0.014107 +- 0.001736	99.21% +- 0.00%	97.67% +- 0.00%
10^{-2}	0.000214 +- 0.000110	0.001031 +- 0.001534	100.00% +- 0.00%	100.00% +- 0.00%
10^{-3}	0.000009 +- 0.000007	0.000069 +- 0.000124	100.00% +- 0.00%	100.00% +- 0.00%
10^{-4}	0.000000 +- 0.000000	0.000001 +- 0.000001	100.00% +- 0.00%	100.00% +- 0.00%
10^{-5}	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
10^{-6}	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
10^{-7}	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
10^{-8}	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
10^{-9}	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
10^{-10}	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%

Divorce para L2

η	***Training MSE***	***Test MSE***	***CCR Train***	***CCR Test***
1	0.021194 +- 0.000810	0.023025 +- 0.000376	97.64% +- 0.00%	97.67% +- 0.00%
10^{-1}	0.015424 +- 0.001866	0.020498 +- 0.000534	97.80% +- 0.31%	97.67% +- 0.00%
10^{-2}	0.007780 +- 0.001360	0.017003 +- 0.001241	98.58% +- 0.31%	97.67% +- 0.00%
10^{-3}	0.001432 +- 0.000591	0.005938 +- 0.001706	99.84% +- 0.31%	98.60% +- 1.14%
10^{-4}	0.000086 +- 0.000049	0.000845 +- 0.001182	100.00% +- 0.00%	100.00% +- 0.00%
10^{-5}	0.000003 +- 0.000002	0.000204 +- 0.000382	100.00% +- 0.00%	100.00% +- 0.00%
10^{-6}	0.000000 +- 0.000000	0.000050 +- 0.000100	100.00% +- 0.00%	100.00% +- 0.00%
10^{-7}	0.000000 +- 0.000000	0.000050 +- 0.000100	100.00% +- 0.00%	100.00% +- 0.00%
10^{-8}	0.000000 +- 0.000000	0.000049 +- 0.000098	100.00% +- 0.00%	100.00% +- 0.00%
10^{-9}	0.000000 +- 0.000000	0.000049 +- 0.000098	100.00% +- 0.00%	100.00% +- 0.00%
10^{-10}	0.000000 +- 0.000000	0.000049 +- 0.000098	100.00% +- 0.00%	100.00% +- 0.00%

noMNIST para L1

η	***Training MSE***	***Test MSE***	***CCR Train***	***CCR Test***
1	0.054875 +- 0.001214	0.052100 +- 0.001604	79.82% +- 1.02%	82.73% +- 1.65%
10^{-1}	0.034871 +- 0.001263	0.031852 +- 0.001660	86.58% +- 0.73%	88.73% +- 0.71%
10^{-2}	0.030462 +- 0.001368	0.028968 +- 0.001372	87.78% +- 0.60%	88.93% +- 0.77%
10^{-3}	0.029876 +- 0.001435	0.029148 +- 0.001345	88.04% +- 0.52%	88.73% +- 0.90%
10^{-4}	0.029820 +- 0.001445	0.029193 +- 0.001339	88.07% +- 0.53%	88.67% +- 0.87%
10^{-5}	0.029815 +- 0.001447	0.029202 +- 0.001339	88.09% +- 0.54%	88.67% +- 0.87%
10^{-6}	0.029814 +- 0.001447	0.029203 +- 0.001339	88.09% +- 0.54%	88.67% +- 0.87%
10^{-7}	0.029814 +- 0.001447	0.029202 +- 0.001338	88.09% +- 0.54%	88.67% +- 0.87%
10^{-8}	0.029814 +- 0.001447	0.029202 +- 0.001338	88.09% +- 0.54%	88.67% +- 0.87%
10^{-9}	0.029814 +- 0.001447	0.029202 +- 0.001338	88.09% +- 0.54%	88.67% +- 0.87%
10^{-10}	0.029814 +- 0.001447	0.029202 +- 0.001338	88.09% +- 0.54%	88.67% +- 0.87%

noMNIST para L2

η	***Training MSE***	***Test MSE***	***CCR Train***	***CCR Test***
1	0.067368 +- 0.001356	0.066200 +- 0.001607	77.71% +- 0.54%	77.47% +- 1.60%
10^{-1}	0.045785 +- 0.001033	0.043954 +- 0.001592	83.24% +- 0.76%	6.67% +- 1.28%
10^{-2}	0.035546 +- 0.001102	0.032802 +- 0.001456	86.40% +- 0.72%	88.80% +- 0.96%
10^{-3}	0.031208 +- 0.001263	0.029187 +- 0.001509	87.73% +- 0.46%	89.07% +- 0.93%
10^{-4}	0.029996 +- 0.001399	0.029073 +- 0.001371	88.04% +- 0.52%	88.87% +- 0.93%
10^{-5}	0.029810 +- 0.001444	0.029208 +- 0.001331	88.09% +- 0.48%	88.60% +- 0.77%
10^{-6}	0.029788 +- 0.001448	0.029221 +- 0.001322	88.11% +- 0.42%	88.60% +- 0.77%
10^{-7}	0.029787 +- 0.001450	0.029233 +- 0.001325	88.11% +- 0.42%	88.60% +- 0.77%
10^{-8}	0.029785 +- 0.001449	0.029233 +- 0.001323	88.11% +- 0.42%	88.60% +- 0.77%
10^{-9}	0.029786 +- 0.001450	0.029231 +- 0.001325	88.11% +- 0.42%	88.60% +- 0.77%
10^{-10}	0.029785 +- 0.001448	0.029237 +- 0.001317	88.11% +- 0.42%	88.60% +- 0.77%

Observando la base de datos de noMNIST podemos ver que a medida que el coeficiente de aprendizaje aumenta, nuestro modelo va aprendiendo, pero cuanto más reducido se hace, menor es la mejora de entrenamiento hasta llegar a un punto donde se mantiene constante. Comparando los dos “tipos” de convergencias podemos ver que el error va tendiendo a 0 para el caso de L1 de manera más óptima que L2. También se observa que el parámetro eta y el tipo de convergencia están relacionados, viéndose que, a medida que el parámetro eta disminuye, la diferencia entre ambos modos va disminuyendo.

3.3 Tercer experimento

Ahora comprobaremos la mejor ejecución tanto de regresión como de clasificación con la inicialización propuesta para el algoritmo KMeans por parte del módulo *Sklearn*. Para ello se modificará este parámetro de la función en nuestro programa python. Se tomará como del ratio aquel valor que mejor resultado ha dado en el primer experimento. El valor de eta será el mismo que le del experimento 1.

Función Seno

KMeans	***Ratio RBF***	***Training MSE***	***Test MSE***
random	***15%***	0.011216 +- 0.000068	0.011216 +- 0.000068
k-means++	***15%***	0.011197 +- 0.000017	0.375319 +- 0.022855

Quake

KMeans	***Ratio RBF***	***Training MSE***	***Test MSE***
random	***5%***	0.028365 +- 0.000063	0.028475 +- 0.000223
k-means++	***5%***	0.028215 +- 0.000051	0.028689 +- 0.000124

Parkinson

KMeans	***Ratio RBF***	***Training MSE***	***Test MSE***
random	***25%***	0.000212 +- 0.000017	0.001064 +- 0.000043
k-means++	***25%***	0.000459 +- 0.000014	0.001186 +- 0.000063

Divorce

KMeans	***Ratio RBF***	***Training MSE***	***Test MSE***	***CCR Train***	***CCR Test***
random	***5%***	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
k-means++	***5%***	0.000001 +- 0.000001	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%

noMNIST

KMeans	***Ratio RBF***	***Training MSE***	***Test MSE***	***CCR Train***	***CCR Test***
random	***50%***	0.000018 +- 0.000002	0.035192 +- 0.001981	100.00% +- 0.00%	87.93% +- 0.49%
k-means++	***50%***	0.029078 +- 0.000597	0.029127 +- 0.000643	87.93% +- 0.31%	88.93% +- 0.83%

En un principio, el módulo *Sticky-learn* comenta que el uso de k-means++ permite obtener mejores resultados a la hora de la aplicación de este algoritmo, pero a efectos de esta práctica, no se han encontrado diferencias apreciables en ello. De hecho, encontramos que en nMNIST, tanto el error de entrenamiento como el accuracy del train, presentan resultados mucho peores.

Para el caso de divorce, se clasifica en todos los experimentos de buena forma para pocas neuronas, presentándose nulas diferencias. Sorprendentemente, para el caso de *seno* y *quake*, los resultados son similares, salvo en el error del test para *quake*.

3.4 Cuarto experimento

Como último experimento se lanzará el programa tomando como problema de regresión los dos datasets de tipo clasificación. Se toma el mejor número de neuronas, y una eta de valor similar al experimento 1.

Divorce

Tipo de problema	***Ratio RBF***	***Training MSE***	***Test MSE***	***CCR Train***	***CCR Test***
Clasificación	***5%***	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%
Regresión	***5%***	0.000000 +- 0.000000	0.000000 +- 0.000000	100.00% +- 0.00%	100.00% +- 0.00%

noMNIST

Tipo de problema	***Ratio RBF***	***Training MSE***	***Test MSE***	***CCR Train***	***CCR Test***
Clasificación	***50%***	0.000018 +- 0.000002	0.035192 +- 0.001981	100.00% +- 0.00%	87.93% +- 0.49%
Regresión	***50%***	0.871545 +- 0.032224	0.920240 +- 0.043723	82.09% +- 0.54%	81.67% +- 0.87%

Soy consciente de que el resultado deberá de presentar una gran diferencia en el CCR tanto en el entrenamiento como en el test, mayor aún en principio para el caso de noMNIST al tener una base de datos mayor. Pero quizá por un cálculo incorrecto del CCR, se muestra que en el caso de divorce no hay una diferencia apreciable. En el caso de noMNIST sí se observa que hay un error altísimo en el entrenamiento y menor porcentaje de acierto en el CCR. Pero como ya comenté antes, deberían apreciarse mucho peores resultados para ambas pruebas.

4. Bibliografía

- Presentaciones y temario de la asignatura.

- Librerías usadas como Sticky-learn, numpy, etc. Todos los enlaces de la documentación se encuentran en el código.

- Python Geek:
https://www.geeksforgeeks.org/python-programming-language/

- Presentacion de Redes de Base Radial:
http://www.varpa.org/~mgpenedo/cursos/scx/archivospdf/Tema5-6.pdf