Optimización de redes neuronales mediante métodos bioinspirados

Antonio Molina García-Retamero

10 de noviembre de 2014

Índice general

Capítulo 1

Introducción

1.1. La propuesta de Trabajo de Fin de Grado

En este documento se recoge la memoria del trabajo de fin de grado que he realizado y que pretende justificar los valores y facultades adquiridas en el estudio y superación de las competencias recogidas en el Grado en Ingeniería Informática. En estas primeras líneas trataré de exponer el trabajo en términos generales y justificar la elección del mismo y las bondades y problemas del mismo.

1.1.1. Justificación y objetivos

En el proceso de elección del proyecto de fin de grado y de los objetivos a plantear para el desarrollo del mismo, he de indicar que primó mi clara vocación investigadora y es por ello que desde el principio le planteé a mi tutor del proyecto mi deseo de realizar algún tipo de investigación básica, con especial interés es el campo de la inteligencia artificial. Es por eso que este proyecto de fin de grado está generalmente enfocado al desarrollo de una investigación que pueda encontrar una aplicación práctica evidente y que cubra en la medida de lo posible todas las competencias adquiridas durante el grado. Por lo tanto, los objetivos generales que cubren este trabajo serían:

- Búsqueda de un problema relativo al campo de estudio sobre el que realizar un proceso investigador con tal de dar solución al problema desde un punto científico-técnico.
- Estudio del estado del arte del problema en cuestión.
- Planteamiento de soluciones.
- Experimentación.

La redacción de esta memoria está estructurada haciendo una división entre el trabajo puramente teórico e investigador que será la primera parte de la misma y que tratará de clarificar la metodología de investigación, el planteamiento del problema objeto de estudio y el estudio del estado del arte. En la segunda parte se tratará el diseño y la implementación del banco de pruebas sobre el que realizar el proceso experimentador y se justificará la elección de las tecnologías y metodologías utilizadas. En

la tercera parte de esta memoria se detallará la experimentación realizada, se recogerán los resultados y se detallarán las conclusiones del trabajo en su conjunto.

Capítulo 2

Proceso de investigación

2.1. Planteamiento del problema

Una vez definido el trabajo y planteados los objetivos de este trabajo fin de grado, el siguiente paso ha sido determinar el problema que se va a tratar dentro del proceso investigador y así tratar de proponer una solución al problema siguiendo una metodología de investigación. Las redes neuronales artificiales han pretendido, desde su concepción, emular, de alguna manera, el proceso cognitivo que se produce en el cerebro de los mamíferos, siempre a una escala reducida, pero sirviéndose de los mismos principios fundamentales. Sin embargo, el problema del aprendizaje es tradicionalmente objeto de la estadísticas y no de la biología. Las redes neuronales artificiales aplicadas al problema de aprendizaje han sido muy estudiadas como un problema estadístico y existe una muy extensa literatura al respecto, sin embargo, no ha sido hasta años recientes que se está abordando el problema del aprendizaje máquina desde un punto de vista biológico. En los últimos años, la neurocomputación ha adquirido una muy notable importancia dentro del campo del aprendizaje máquina y propone sistemas que realmente pretenden emular a los sistemas biológicos y en los que la posibilidad de llegar a entender quizá un poco más los procesos que dotan a los sistemas biológicos de inteligencia y quizá incluso emular algunas de sus funcionalidades es un campo muy prometedor y cargado de retos ilusionantes.

Partiendo del problema de tratar de emular mecanismos biológicos en los modelos de aprendizaje máquina, mi tutor, Daniel Ruiz, me mostró el trabajo de Diego Andina sobre la aplicación de la metaplasticidad neuronal al perceptrón multicapa (MLP) y me propuso aplicarlo a otro tipo de redes como las RBFNN.

2.2. El estado del arte

Dado que partimos de un trabajo en curso, que es la implementación de la metaplasticidad artificial en redes neuronales artificiales, el estudio del estado del arte ha sido un trabajo aurduo que ha requerido de un estudio bastante en profundidad de las diferentes disciplinas bajo las que se contempla el problema que hemos definido y que cubren desde la neurobiología a la estadística y el aprendizaje

máquina. Es por esto que me serviré de esta sección primero para demostrar de alguna forma el trabajo realizado en la documentación y adquisición de conocimientos necesarios para comprender el problema y plantear una solución así como para tratar de introducir al lector, de la forma más amena posible, en los conceptos y terminología en que se definen la problemática y la solución propuesta al problema.

El trabajo se inicia con el estudio de lo relativo a la metaplasticidad en las redes neuronales biológicas. Además, he dedicado tiempo a estudiar y entender los modelos estadísticos bajo los que se estudian las redes neuronales artificiales para buscar la forma de aplicar este principio, que es la metaplasticidad artificial, a otros tipos de redes neuronales artificiales más allá, y partiendo, del trabajo que ha desarrollado Diego Andina en las MLP.

Una vez adquiridos los conocimientos necesarios para comprender el problema, se realizará un estudio del estado del arte de las diferentes técnicas de las RBFNN con tal de encontrar la forma de aplicar el concepto de metaplasticidad que se da de forma natural en las redes neuronales biológicas.

En las siguientes subsecciones se definirán primero la metaplasticidad en términos biológicos y se expondrá la propuesta de Diego Andina para poder especificar el problema y, partiendo de ahí, plantear la solución propuesta.

En este punto de la investigación se ha recurrido de nuevo a la bioinspiración tratando de encontrar una perspectiva bajo la que contemplar las RBFNN y sobre la que plantear una solución al problema.

2.3. La metaplasticidad en redes biológicas

Con el término metaplasticidad nos referimos al cambio dependiente de la actividad que modula la plasticidad sináptica en los sistemas neuronales. Estos cambios en la plasticidad pueden ser del tipo LTP[1] y LTD[2]. Abraham and Bear lo definió de forma sencilla como: *plasticity of synaptic plasticity*, es decir, la plasticidad de la plasticidad sináptica. A diferencia de los mecanismos convencionales de neuromudalación de la plasticidad sináptica en los que son neurotransmisores y hormonas los que inducen cierto grado de LTP y LTD, la metaplasticidad se refiere al cambio que es provocado en un momento dado por lo que comunmente denominamos *primar* la actividad. Es decir, la metaplasticidad se refiere a la inducción en los cambios de los LTP y LTD derivados de la actividad. Reforzando aquellas conexiones sinápticas relativas a las *experiencias* más comunes y deprimiendo aquellas relativas a *experiencias* menos comunes.

Funcionalmente, la metaplasticidad dota a las conexiones sinápticas de la capacidad de integrar señales relativas a la plasticidad a través del tiempo. Además, la metaplasticidad tiene un importante papel como regulador de los umbrales de los LTP y LTD para mantener las conexiones sinápticas en un rango funcional dinámico que evite que estas conexiones se vuelvan demasiado fuertes o demasiado débiles como consecuencia de los mecanismos de plasticidad sináptica convencionales basados en

[1] **Long-Term Potentiation** es la potenciación de la eficiencia provocada por los eventos más reciente en la señal transmitida entre dos neuronas que se estimulan síncronamente

[2] **Long-Term Depression** es la reducción de la eficiencia provocada por los evento menos reciente en la señal transmitida entre dos neuronas que se estimulan síncronamente

neurotransmisores y hormonas.

2.4. Metaplasticidad en redes neuronales artificiales

El doctor Diego Andina realizó una aproximación a la integración de los mecanismos de metaplasticidad en ..., donde se propone un mecanismo de metaplasticidad artificial (AMP[3]) aplicado a las MLP[4]. En las siguientes líneas trataré de explicar la interpretación que Diego Andina propone sobre la metaplasticidad en el cálculo del error y de la modificación del algorítmo de *backpropagation* propone para la implementación de la AMP en los MLP.

El problema del entrenamiento en cualquier problema de aprendizaje es el de minimizar una función de error ajustando los parámetros del modelo en base a un conjunto de vectores de entrada $x = (x_1, x_2, ..., x_n), (x \in \Re^n)$ que conforman el *training set*. En base al criterio de error $E(x)$ que elijamos, el error esperado del sistema será por tanto:

$$E_M = \int_{\Re^n} E(x)f(x)dx = \int_{\Re^n} e(x)dx \tag{2.1}$$

Sobre lo que podemos hacer la siguiente transformación:

$$E_M = \int_{\Re^n} \frac{e(x)}{f_X^*(x)} f_X^*(x)dx = \varepsilon^*\{\frac{e(x)}{f_X^*(x)}\} \tag{2.2}$$

y calcular el error del modelo sirviéndonos del siguiente estimador:

$$\widehat{E}_M = \frac{1}{M}\sum_{k=1}^{M} \frac{e(x_k^*)}{f_X^*(x_k^*)} \tag{2.3}$$

donde $x_k^*, k = 1, 2, ..., M$ son muestras independientes cuya *pdf* es $f_X^*(x)$. Como se observa, $f_X^*(x)$ es la distribución de probabilidad de la función que tratamos de generalizar con nuestro modelo a través de nuestras muestras de entrenamiento. Por lo tanto, idealmente,

$$(f_X^*(x))_{opt} = \frac{1}{E_M} e(x) \tag{2.4}$$

Lo que se pretende conseguir a través de la introducción de la función subóptima de las ecuaciones 2.2 y 2.3 es incluir el factor de probabilidad a posteriori de la muestra con tal de provocar un efecto de LTP o LTD en los pesos sinápticos consiguiendo así introducir un mecanismo de metaplasticidad en la red neuronal artificial. Como se expresa en la ecuación 2.4, la función $f_X^*(x)$ óptima sería la *pdf* de la señal de entrada de nuestro modelo ($f_X(x)$), lo que es desconocido por nosotros, con lo que, en la práctica, se elige una $f_X^*(x)$ arbitraria tal que $f_X^*(x) \simeq f_X(x)$.

La figura 2.4 ilustra como se incorpora el mecanismo de metaplasticidad como un factor de corrección sobre los pesos. Como se observa, esta corrección del error se hace en base a la salida esperada siendo esta, como ya hemos dicho, $f_X^*(x)$.

[3]Artificial Metaplasticity
[4]Multilayer Perceptron

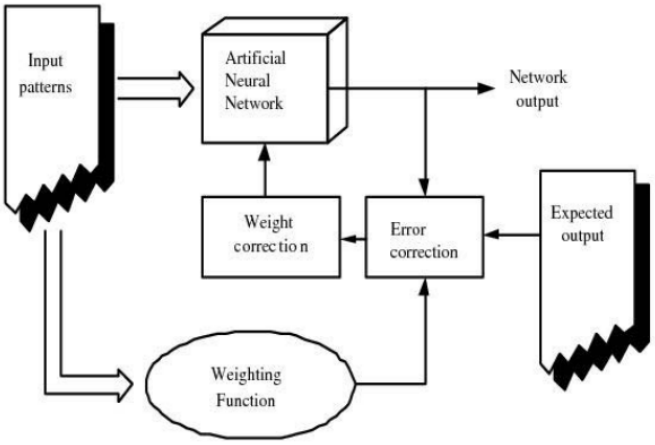

Figura 2.1: Iteración en el entrenamiento de una red neuronal aplicando el mecanismo de metaplasticidad

2.4.1. Propuesta de implementación de la AMP en las MLP

En base a esta interpretación que Diego Andina hace de la metaplasticidad, en el mismo artículo (cita) se propone la siguiente modificación sobre el algoritmo de *backpropagation* para incluir la función subóptima $f_X(x)$ a los MLP.

$$\frac{\partial \varepsilon(W)}{\partial w_i^{(S)}} = \frac{\partial}{\partial w_i^{(S)}} \left(\frac{1}{2} \frac{(y - \widehat{y}^{(S)})^2}{f_X^*(x)} \right) = \frac{1}{f_X^*(x)} \frac{\partial \varepsilon(W)}{\partial w_i^{(S)}} \tag{2.5}$$

$$\delta_j^{(S)} = (y_j - \widehat{y}_j^{(S)}) \cdot \frac{f'^{(S)}}{f_X^*(x)} \tag{2.6}$$

siendo s el contador de capas, $s = 1, 2, ..., S$, j e i son el contador de nodos y entradas y $\delta_j^{(S)}$ el error de un nodo j para la capa s. Por tanto, la ecuación 2.6 define el cálculo del error para los nodos de la última capa. Para el resto de capas, el error se propaga siguiendo la forma convencional del *backpropagation*.

De todo lo comentado anteriormente, queda patente que una de las claves en la implementación de la metaplasticidad a las redes neuronales es la elección acertada de la función subóptima $f_X(x)$. En el caso concreto de las MLP, para clasificar L clases H_l para $l = 0, 1, ..., L-1$, y basándonos en el teorema de Bayes, obtenemos

$$\widehat{y}_l \simeq P(H_l/x) = \frac{f_X(x/H_l) \cdot P(H_l)}{f_X(x)} \tag{2.7}$$

si asumimos la salida de nuestra red $\widehat{y}_l$ como $P(H_l/x)$, la propia salida de la red neuronal generaliza $f_X(x)$ con lo que esta puede ser utilizada como función subóptima.

De esta forma, Diego Andina obtiene los resultados de la figura 2.4.1 y que representa la media del error para redes con la modificación propuesta (BPW) para un número creciente de operaciones en contraposición con la media del error en redes entrenando con un algorítmo de *backpropagation* tradicional(BP). En este caso, la función de distribución de probabilidad, se ha asumido la siguiente

gausiana

$$f_X^*(x) = \frac{A}{\sqrt{(2\pi)^n} \cdot e^{\frac{B}{8} \sum_{i=1}^{n} x_i^2}} \tag{2.8}$$

donde A y B son los parámetros de la gausiana que en este caso representan los parámetros de la metaplasticidad. En el artículo (cita) se muestra el siguiente ejemplo de como puede mejorar esta propuesta para una entrada concreta en un escenario, se supone, ideal:

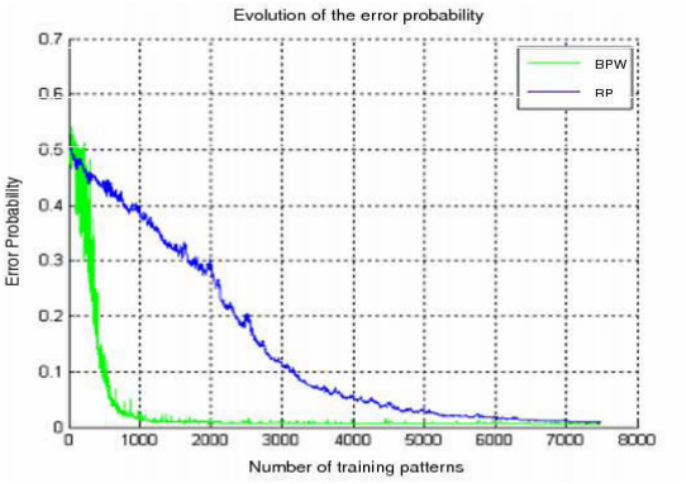

Figura 2.2: Comparación de la evolución del aprendizaje del algorítmo ponderado (BPW) frente al tradicional (BP)

Implementación propia

Sobre esta propuesta de modificación del algorítmo de *backpropagation* realicé mi propia implementación con tal de poder hacer mi propia experimentación. Esta implementación la basé en el desarrollo de una MLP que realicé para la asignatura de *Razonamiento Automático* en C++. El objetivo de esta implementación era tan solo el de poder experimentar de primera mano, utilizar diferentes funciones subóptimas y diferentes datos de entrenamiento. Así, en la siguiente subsección expondré las conclusiones derivadas del estudio y la experimentación de la técnica propuesta por Diego Andina. Esta implementación está publicada en el siguiente repositorio de GitHub[5].

Esto creo que se me queda corto, pero tampoco se muy bien que más decir

Conclusiones del estudio

Una vez asimilados los conceptos y habiendo realizado una implementación y haber experimentado primero sobre el conjunto de datos sobre los que se publica el artículo y después con otros conjuntos de datos públicos y típicos en los problemas de aprendizaje máquina y habiendo experimentado con diferentes funciones subótimas, he alcanzado las siguientes conclusiones

[5] https://github.com/aydevosotros/AMMLP

- Encontrando una función subóptima que se aproxime a la función de distribución de probabilidad de los datos que componen la señal de entrada podemos obtener una gran mejora emulando los procesos LTP y LTD de los sistemas biológicos.
- A priori, $f_X(x)$ es desconocida en los problemas de aprendizaje máquina, con lo que escoger una $f_X^*(x)$ subótima apropiada es un problema muy considerable.
- La posibilidad de aproximar $f_X(x)$ por la salida de la red neuronal $\widehat{y}_l$ cuando esta está lo suficientemente entrenada como para proveer una aproximación lo suficientemente buena puede ser muy interesante.

2.5. Metaplasticidad Artificial en RBFNN

En esta sección trataré de expresar la problemática observada en el proceso de encontrar y aplicar algún mecanismo de metaplasticidad artificial aplicado a las redes neuronales con función de activación de base radial (RBFNN). Para ello, haré en primer lugar una breve introducción a las RBFNN tratando de introducir al lector en las principales características de este tipo de redes para después exponer la problemática encontrada.

2.5.1. Características de las RBFNN

Arquitectura de las RBFNN

Características principales

Metodologías de entrenamiento

2.5.2. Aplicación directa del estimador de probabilidad

Una primera implementación de la metaplasticidad a las RBFNN podría ser la de incluir la estimación de probabilidad subóptima en la fase de entrenamiento de pesos con tal de ponderarlos de forma similar a, como explicado, Diego Andina propuso para las AMMLP.

2.5.3. Problemática

Una vez asumido el concepto de metaplasticidad en las redes neuronales biológicas y partiendo de la propuesta de metaplasticidad artificial en las MLP de Diego Andina, estamos en disposición de abordar el problema. Como ya se ha expuesto anteriormente y en esencia, la propuesta de Diego Andina se basa en la emulación de los procesos LTP y LTD añadiendo un factor que pondera la derivada parcial en el descenso por gradiente de la muestra con respecto al peso de la conexión sináptica que representa cuan *raro*, en términos de probabilidad, es la muestra respecto a la clase en que se clasifica a posteriori. Esta probabilidad a posteriori se estima por la propia salida de la red neuronal en ese estado del entrenamiento.

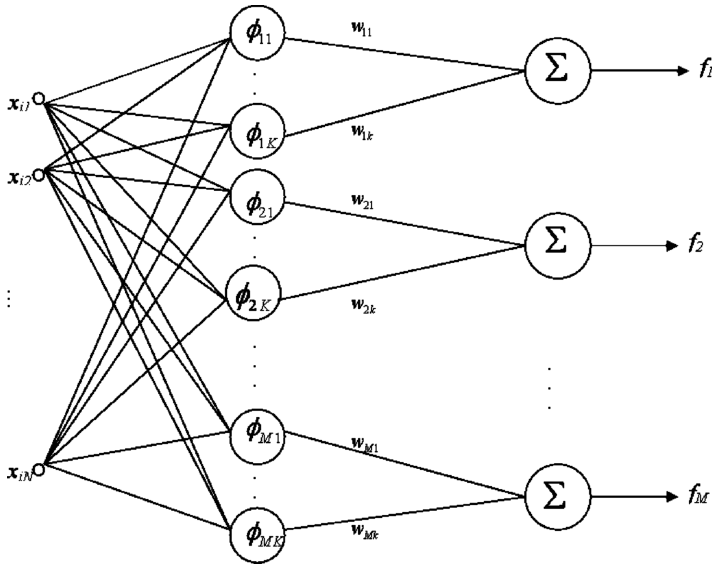

Figura 2.3: Arquitectura de las RBFNN

En este punto, queda patente que existe numerosos problemas a la hora de aplicar esta técnica a otro tipo de redes neuronales, en particular en las RBFNN. En la figura 2.5.3, se representa la arquitectura típica de una RBFNN.

Un problema fundamental se debe a la salida de la última capa de sendas redes neuronales. Para implementar un mecanismo de metaplasticidad similar al descrito en las AMMLP, se requiere de una estimación de la probabilidad de pertenencia a la clase (cuan *rara* es la muestra) y en las AMMLP la estimamos como la salida de la propia red neuronal. Ya que la neurona de la última capa es una función logística [6], asumimos ésta como la probabilidad a posteriori de pertenencia a la clase. Por su parte, la salida de una RBFN para el problema de clasificación se calcula como $sign(f(x) = \sum w\phi(x))$. Queda patente que la salida de la RBFNN no nos sirve para estimar una probabilidad a posteriori de la muestra.

Sin embargo, la propia naturaleza de las RBFNN parece resultar idónea a su vez para experimentar con técnicas. Este *parecer* es una percepción completamente intuitiva, pero existen ciertas características que trataré de explicar a continuación que pueden hacer de estas redes idóneas para representar las redes neuronales biológicas y a la forma en que estas representan la información y que me inspiraron a plantearme el problema en multitud de formas tratando de encontrar la forma de encajar estas características con la naturaleza del aprendizaje y de la metaplasticidad. Estas características son, en líneas generales:

- La capa oculta característica de las RBFNN está compuesta por *neuronas* cuya función de activación es típicamente una Gausiana en la forma . Estas gausianas tienen dos parámetros básicos y que son fundamentalmente dependientes de la naturaleza de la señal de entrada y que son un centroide y la varianza.

[6] Una función logística es una función contínua cuya salida $\in [01]$. Un ejemplo de función logística típia es $f_{log}(z) = \frac{1}{1+e^{-z}}$

- La capa oculta, a su vez, realiza una transformación no lineal en un espacio basado en la distancia. Esta característica sirve de nexo para el estudio de las RBFNN dentro del conjunto de los *Kernel Methods*.
- La salida de la red neuronal, con la forma $f(x) = \sum w\phi(x)$ es una transformación lineal sobre la salida de las gausianas.

2.6. Ampliación del estado del arte

Expuestos los problemas a los que nos enfrentamos

2.6.1. Tonotopía y Retinotopía

2.6.2. Kernel Methods

2.7. Propuesta de solución

En base a los trabajos previos, la idea de servirnos de la naturaleza de los datos de nuestro entrenamiento para hacernos una idea a priori de cuan significativa es una muestra y actuar ponderando esa çonexión sinápticaçoncreta parece encajar de forma especial en el concepto de RBFNN debido a la naturaleza de las Funciones de Base Radial. Quiero decir, las Funciones de Base Radial nos proporcionan una transformación no lineal que nos permite encontrar un hiper-plano que en algún espacio multidimensional nuestros datos serán separables. El modelo de las RBFNN, como ya hemos comentado, se sirven de un modelo con la forma $f(x) = \sum w\phi(x)$ donde $\phi(x)$ es un función de base radial. Este tipo de funciones pueden ser muy variadas y se ha experimentado con multitud de ideas alrededor de esto sin embargo en los casos más habituales se utiliza una función gausiana en donde tenemos que inferir tanto la media como los centroides para cada una de las funciones en nuestro training set. De esto se deriva un entrenamiento de dos fases en el que en la primera fase inferimos los datos necesarios para calcular la activación de la capa oculta para hacer luego encontrar los parámetros para la transformación lineal sobre la salida de esta capa en una segunda fase de entrenamiento.

Aquí es donde pretendo encontrar y probar diferentes formas de optimizar la segunda fase de entrenamiento sirviéndonos del estudio estadístico que implica la primera fase. Esto es, Cuando en la primera fase de entrenamiento buscamos los centros para nuestras funciones de activación obtenemos información relevante que podemos utilizar para estimar, como hablábamos antes, como considerar una probabilidad de pertenencia a una clase si consideramos el centroide como una clase, etc... Multitud de ideas se pueden considerar y en última instancia lo que se estaría haciendo sería entremezclar de diferentes formas el proceso de aprendizaje no supervisado a priori para influir en el proceso de aprendizaje supervisado de la segunda fase.

Capítulo 3

Implementación

Tendría que encontrar el sitio donde poner qué se ha implementado y por qué

3.1. Metodología de desarrollo

Aquí comento que he realizado la implementación siguiendo el paradigma TDD para la implementación de las redes.

3.2. Lenguaje y herramientas

3.3. Implementación de las redes

3.4. Pruebas y métricas de rendimiento

Dado que por medio de este trabajo se pretende la demostración de la mejora propuesta sobre las RBFNN, estableceré una serie de pruebas que pretenderán poner a prueba las diferentes implementaciones de las redes para resolver problemas típicos de aprendizaje máquina. En secciones anteriores hemos comentado la problemática y las diferentes alternativas propuestas para entrenar las RBFNN. La batería de pruebas es la parte probablemente más crítica del proyecto y con especial esmero he diseñado un conjunto de pruebas que pretende ser exhaustivo y con el que pretendo primero valorar el comportamiento de las soluciones típicas existentes y las mejoras propuestas en esta investigación.

De entre los escenarios típicos del aprendizaje máquina podemos hacer dos grandes grupos: tal y clasificación. Para este proyecto he seguido la estrategia de construir, para sendos tipos de problemas, sets de datos aleatorios cuyos parámetros intrínsecos puedan ser manipulados fácilmente permitiendo generar de forma sencilla tantos escenarios como se crean apropiados para los diferentes test. En busca de evitar el "data snoopingz una vez determinadas las configuraciones para los que los diferentes tipos de redes tienen comportamientos interesantes, las redes serán testeados sobre sets de datos estándares en la investigación sobre aprendizaje máquina.

- Reconocimiento de patrones
  - Onda compuesta aleatoria
  - Otra idea que se me ocurra
- Clasificación
  - Clasificación sobre datos aleatorios artificiales
  - Clasificación sobre datos reales

En las siguientes secciones trataré de clarificar todo esto para cada tipo de pruebas. Explicaré las cosas fundamentales de la implementación tratando de no perder al lector en el código pero dándo una visión clara y precisa de lo que se desea poner a prueba con cada test. Sin embargo, antes de entrar en detalle sobre la construcción de los conjuntos de datos me gustaría dejar patente el guión básico que sigo a la hora de diseñar las prubas.

Anteriormente hemos discutido las fases del entrenamiento de redes RBFNN y de las soluciones típicas. Además discutimos acerca de como podemos implementar la intuición de la metaplasticidad en la fase supervisada a partir de la información estadística que supone el aprendizaje no supervisado de la primera parte del entrenamiento en busca de los parámetros de las funciones de base radial.

Con todo esto, una primera batería de test tratará de determinar en qué condiciones, el cálculo de la pseudoinversa deja de resultar rentable en términos de coste computacional con respecto a al descenso por gradiente. Este coste computacional es directamente proporcional al tamaño del set de datos, con lo que esta prueba nos dará una estimación de cuan grande ha de ser al menos nuestro conjunto de datos para que resulte rentable la aproximación del descenso por gradiente que es necesario para la implementación de la intuición de metaplasticidad aquí propuesta.

A continuación, someteremos a una batería exahustiva de test generados artificialmente y sobre los que variaré los parámetros de los diferentes tipos de redes y de los propios conjuntos de datos en busca de las condiciones en que cada tipo de RBFNNs de los propuestos ofrece un comportamiento mejor que el resto. Con esto podremos determinar si existen condiciones en las que, en efecto, la propuesta de metaplasticidad supone una mejora sobre las aproximaciones existentes.

Por último, se realizarán pruebas sobre sets de datos conocidos y ampliamente utilizados en el campo del aprendizaje máquina. Con esto se pretende evitar la aparición de resultados inválidos por la propia intervención y manipulación de los parámetros y datos en busca de las condiciones en las que la esta implementación resulte mejor además de poder ofrecer comparativas que puedan ser reconocidas por quienes trabajan en otras propuestas sobre los mismos conjuntos de datos.

3.4.1. Pruebas de clasificación

Uno de los usos más habituales de las RBFNN es cuando tenemos que clasificar sobre un conjunto de datos cuyas clases no son linealmente separables. Anteriormente hablamos de Ya que, a través de las RBFNN, obtenemos un modelo con la forma $f(x) = \sum w_i \phi(r)$ (que es caracterizado con funciones de base radial) podemos aproximar funciones complejas cuya salida sigue una distribución gaussiana

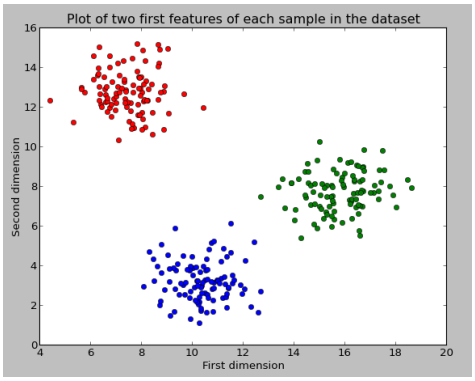

Figura 3.1: Conjunto de datos con distribución normal

(la más frecuente con mucha diferencia de entre las variables aleatorias que se dan en la naturaleza) con notable precisión. La figura 3.4.1 es un ejemplo de datos generados con el generador de datos clusterizados que he implementado. Este recibe como parámetros básicos el número de centroides y el número de muestras y genera estos datos de forma aleatoria tomando como centro los k-centroides también tomados aleatoriamente. En las siguientes líneas introduciré esta clase sin entrar en excesivo detalle de la implementación pero sí tratando de aclarar las cuestiones que considero claves.

Generación de datos aleatorios

Para este tipo de test necesitamos generar datos que se distribuyan de forma normal alrededor de un centroide. Una distribución normal es aquella que cuya función de densidad de probabilidad queda definida por la media (μ) y la desviación estándar (σ) del conjunto de datos (x) de la siguiente forma:

$$pdf(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\left(-\frac{(\mu-x)^2}{2\,\sigma^2}\right)}$$

Existen multitud de generadores aleatorios de este tipo de datos. En este caso utilizaré el propio de NumPy como muestro a continuación para generar las n muestras por cada centroide ya que este definirá la media. La desviación estándard será proporcionada como parámetro en el constructor de la clase y nos permitirá jugar con la forma que adoptarán los datos generados.

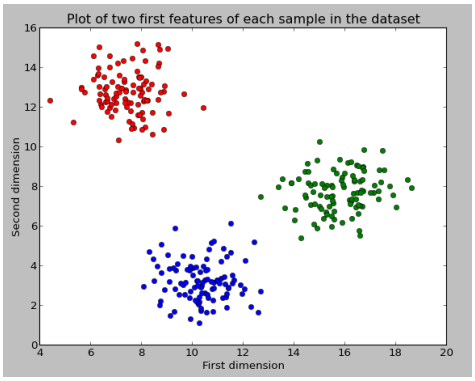

Figura 3.2: Conjunto de datos con distribución normal

3.5. Implementación básica de una RBF

3.6. Extendemos con centróides K-NN

Capítulo 4

Resultados

4.0.1. Pruebas de clasificación

Ya hemos comentado anteriormente como he desarrollado el generador de datos aleatorios y como este evalúa el error del clasificador. En este proceso de evaluación consideraré una serie de conjuntos de datos con diferentes números de categorías que siguen una distribución normal en un número arbitrario de dimensiones. Sobre cada uno de estos conjuntos de datos evaluaré el funcionamiento de las distintan RBFNN estudiadas con diferente número de clústeres (neuronas con activación en base a una función de base radial en la primera capa de la red) evaluando tanto la clasificación (aciertos, fallos, falsos positivos, falsos negativos, preción y recall") como los tiempos de entrenamiento.

4.1. Comparativa de RBFNN con y sin metaplasticidad

4.2. Otros resultados de interés

RBFNNs performance over n samples						
nC	Random centroids		Knn		Metaplasticity	
	Time	Accuracy	Time	Accuracy	Time	Accuracy
10	1	2	1	2	1	2
10	1	2	1	2	1	2