Trabajo Fin de Máster

en Ingeniería Electrónica, Robótica y Automática

Desarrollo de nuevas estructuras neuronales para la implementación de Deep Learning.

Autor: Ing.Gerardo Gómez Pérez

Tutor: Prof. José Manuel Quero Reboul

Dpto. Ingeniería Electrónica Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2020

Trabajo Fin de Máster

Máster en Ingeniería Electrónica, Robótica y Automática

Desarrollo de nuevas estructuras neuronales para la implementación de Deep Learning.

Autor:

Ing. Gerardo Gómez Pérez

Tutor:

José Manuel Quero Reboul

Catedrático de Universidad

Dpto. Ingeniería Electrónica

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2020

Trabajo Fin de Máster: Diseño de nuevos modelos de sinapsis para redes neuronales artificiales.

Autor: Ing. Gerardo Gómez Pérez
Tutor: Prof. José Manuel Quero Reboul

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2020

El Secretario del Tribunal

AGRADECIMIENTOS

A mis padres y abuelos, por su sacrificio, su confianza en mí y su aliento.

A los amigos que me acogieron y ayudaron en este empeño.

A mis profesores, en especial a mi tutor José M. Quero, por su profesionalidad y entrega.

A Ceidis, por su permanente compañía.

A todos, gracias.

RESUMEN

En este Trabajo Fin de Máster se definen y desarrollan nuevos modelos matemáticos de sinapsis para su incorporación en redes neuronales artificiales con el objetivo de dar solución a problemáticas de asociación de patrones y su clasificación.

El documento comenzará con una descripción del proyecto general mostrando conocimientos generales del diseño tradicional de redes neuronales artificiales tipo perceptrón para comprender su funcionamiento y dar una mayor claridad, previa a su posterior modificación, al sustituir el modelo de sinapsis estándar por estos nuevos modelos.

Luego se describe cada modelo de sinapsis exponiendo su interpretación geométrica y características principales. También se analizarán las respuestas de la sinapsis estándar ante casos críticos de valores de entrada seleccionados que conllevan a salidas indeseadas y cómo se logran corregir con cada uno de estos nuevos modelos.

Para medir la capacidad de aprendizaje de redes conformadas con estos nuevos modelos se implementó un desarrollo genérico del algoritmo Backpropagation para redes multicapas mediante entrenamiento supervisado.

Por último, se estudia las prestaciones de redes neuronales modificadas con estos modelos de sinapsis ante una problemática de reconocimiento, de dígitos escritos a mano, utilizando la base de datos MNIST. Además, se realiza una comparación final del desempeño de cada red, destacando aspectos del proceso de entrenamiento y tasa de aciertos con el conjunto de validación.

Los mayores problemas encontrados durante el desarrollo del proyecto fueron, en primer lugar, el carácter novedoso de la investigación que dificultó la obtención de conocimientos y resultados previos sobre el tema y, por otro lado, la dificultad del ajuste de los parámetros de diseño de las redes neuronales, por lo que se optó por la utilización de métodos de diseño de experimentos, permitiéndonos optimizarlas y compararlas de la forma más justa posible.

Palabras clave

Redes Neuronales Artificiales; modelos de sinapsis; backpropagation.

ABSTRACT

This Master's Final Project defines and describes new mathematical models of synapses for their incorporation into Artificial Neural Networks (ANN) in order to solve pattern association's problems and their classification.

The document begins with a description of the general project showing general knowledge of the traditional design of perceptron-type ANN to understand its operation and provide greater clarity, prior to its subsequent modification, by replacing the standard synapse model with these new models.

Then each synapse model is described while exposing its geometric interpretation and main characteristics. The standard synapse's responses to critical cases of selected input values that lead to unwanted outputs will also be analyzed, as well as how they can be corrected with each of these new models.

To measure the learning capacity of networks built with these new models, a generic development of the Backpropagation algorithm for multilayer networks was implemented through supervised learning.

Finally, the performance of the modified ANNs with these synapse models is studied in the application of a recognition problem of handwritten digits while using the MNIST database. Moreover, a final comparison of the performance of each network is made while highlighting aspects of the training process and hit rate with the validation set.

The main problems encountered during the development of the project were, first, the novel nature of the research that made it difficult to obtain knowledge and previous results on the subject and, secondly, the difficulty of adjusting the design parameters of the neural networks, which is why the use of experimental design methods was chosen, allowing us to optimize and compare them in the fairest way possible.

Keywords

Artificial neural networks; synapse models; backpropagation.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

OBJETIVOS

En el Departamento de Electrónica de la Escuela Técnica Superior de Ingeniería se está llevando a cabo un proyecto de investigación y desarrollo de nuevas estructuras neuronales para la implementación de Deep Learning, dentro del proyecto "Integración de opto-electroestimulación y procesamiento de señales para el estudio in-vitro de larga duración del comportamiento neuronal" (RTI2018-100773-B-C33), financiado por el Ministerio de Ciencia, Innovación y Universidades. Se propone este trabajo fin de máster que tiene como objetivo comprobar el comportamiento de nuevos diseños mediante la realización de experimentos en diversos problemas de aplicación real.

Este Trabajo Fin de Máster se centra en el diseño y comprobación de nuevos modelos matemáticos de sinapsis para redes neuronales artificiales, basados en conceptos diferentes al modelo estándar y, aunque más alejados de su inspiración biológica, presentan características idóneas para su implementación en aplicaciones de clasificación.

MOTIVACIÓN DEL PROYECTO

Una red neuronal artificial (RNA) es una tecnología computacional que puede ser utilizada en diversas aplicaciones y para la resolución de problemas complejos de ciencia, ingeniería y negocios. Las redes neuronales se pueden desarrollar en un tiempo razonable y pueden realizar tareas específicas mejor que otras tecnologías convencionales, con una alta tolerancia a fallos y con un alto grado de paralelismo en el procesamiento de los datos. La importancia de estas se refleja en que pueden resolver problemas como los de visión o aprendizaje, en los que, si se desea obtener una respuesta en tiempo real, resulta imprescindible procesarlas en paralelo.

Las RNA están causando un gran impacto, captando la atención de los profesionales debido a su extraordinaria aplicabilidad práctica. Las redes neuronales se utilizan para resolver problemas estadísticos y de análisis de datos, como la clasificación de patrones, análisis financiero, control predictivo y optimización de sistemas. Tienen la poderosa capacidad de inferir significados a partir de datos complejos o inexactos, con gran robustez y no linealidad.

Todo esto conlleva a destacar la importancia que tiene el estudio de este campo, que motivó a llevar a cabo la línea de investigación en redes neuronales por el Departamento de Electrónica de la Escuela Técnica Superior de Ingeniería, en los últimos años. Una gran parte de esta investigación en torno al desarrollo de nuevas estructuras neuronales artificiales para su aplicación en aprendizaje profundo está descrita en este trabajo.

ESTRUCTURA DEL DOCUMENTO

Este trabajo está estructurado en cuatro capítulos, en el primero realizamos una introducción al contenido en general sobre el que se desarrolla esta investigación, resaltando principalmente el modelo de neurona estándar a modificar, específicamente su modelo de sinapsis, cuyas deficiencias y problemas se detallan también en un apartado del capítulo.

En el segundo capítulo se describen los nuevos planteamientos para modelo de sinapsis, se expresan las interpretaciones geométricas de los modelos y se analizan sus respectivas salidas ante posibles casos críticos de valores de entradas.

El capítulo 3 recoge una generalización del algoritmo de aprendizaje backpropagation, como primer paso para lograr un entrenamiento de las redes neuronales utilizadas y medir su funcionalidad. Posteriormente es particularizado el algoritmo para cada modelo, se plantean los detalles de la base de datos utilizada como elemento básico para un entrenamiento supervisado y se describen las arquitecturas de las redes neuronales a implementar.

Para concluir, en el cuarto capítulo se realizan las simulaciones y se muestran los resultados obtenidos tanto para el proceso de aprendizaje, en el cual se incluye el análisis del comportamiento de los más influyentes hiperparámetros y la optimización de las redes neuronales, como en el posterior proceso de reconocimiento de patrones en dos escenarios, sin ruido en las imágenes o con presencia del mismo, teniendo en cuenta además el factor de selectividad presente en los modelos de sinapsis.

MATERIALES Y MÉTODOS

Para establecer un estudio de cada modelo de sinapsis planteado en este trabajo, se conformaron cuatro redes neuronales, cada una con su desarrollo matemático, incluyendo procesamiento y aprendizaje. Estas RNAs fueron optimizadas para comparar sus rendimientos aplicándolas sobre la base de datos MNIST de dígitos manuscritos. Para ello se realizó una búsqueda sistemática para cada parámetro de las redes, seleccionando una muestra de valores de los mismos y comparando sus resultados en el entrenamiento y posterior reconocimiento con un nuevo conjunto de patrones. La regla del mínimo error cuadrático fue la utilizada para cuantificar el error global cometido en cualquier momento durante el proceso de aprendizaje.
En la parte de simulación de la investigación, desde el enfoque software, para el entrenamiento y operación de las redes, la codificación y obtención de resultados se realizó en Matlab R2020a de MathWorks en un computador tipo laptop con un Procesador Intel (R) Core (TM) i7-8550U CPU @ 1.80GHz de 8 núcleos y memoria RAM de 16GB. Destacar que no se hizo uso de ninguna toolbox o herramienta para redes neuronales que brinda Matlab, por lo que se implementó la programación para la operación y entrenamiento de las redes, además de concebirlas dentro del programa.

CAPITULO I. INTRODUCCION

Aunque se suponía, en sus inicios, que la transmisión de información mediante impulsos nerviosos entre neuronas biológicas tenía lugar en contactos especializados llamados sinapsis, presentes en las fibrillas nerviosas terminales de las neuronas; no se conocía la naturaleza de la señal transmitida. Se manejaba la hipótesis de que la terminación presináptica expulsaba un neurotransmisor hacia la membrana de la célula postsináptica, y esta producía como resultado un potencial de acción. Posteriormente fue descrito más a detalle este proceso por Bernard Katz quien corroboró que las terminaciones de las células nerviosas contienen pequeñas vesículas sinápticas que expulsan el neurotransmisor en forma de señales químicas o eléctricas.

Estas sinapsis forman parte de un sistema mucho más complejo como es el cerebro humano, encargado de procesar toda la información. Desde mediados del siglo XX, se han desarrollado modelos computacionales que intentan simular estos procesos biológicos [1]. Uno de estos modelos computacionales es el de las redes neuronales artificiales (RNA).

Las RNA o sistemas conexionistas se componen de múltiples nodos interconectados entre sí que imitan las conexiones entre las neuronas, capaces de tomar datos de entrada y realizar operaciones simples. La arquitectura de los sistemas de RNA se diferencia de la arquitectura convencional Von Neumann, presente en la mayor parte de los ordenadores actuales, en su procesamiento de información de forma paralela y no secuencial. Los sistemas de procesamiento en paralelo presentan una serie de ventajas como la robustez en su funcionamiento cuando una pequeña parte del sistema resulta defectuosa, ya que la fiabilidad del sistema depende de la interacción paralela de todas las unidades, y no de cada unidad que compone una larga cadena de operaciones en el sistema convencional y que puede estropear totalmente la computación.

1.1 MODELO ESTANDAR DE NEURONA

La estructura y funcionamiento de la unidad elemental de procesamiento de una red conexionista es muy sencilla. Este es el ejemplo de la neurona artificial llamada perceptrón introducida en 1958 por Frank Rosenblatt que realiza ciertos cálculos para detectar características en los datos de entrada. Su modelo estándar [2] plantea que la entrada O_i^{L-1},

señal de salida de la neurona *i* de la capa *L-1* que excita a la neurona *j*, es ponderada proporcionalmente por su asociado peso sináptico W_{ij}, que se encarga de reforzar o debilitar la señal simulando estímulos excitatorios o inhibitorios según corresponda y luego se realiza la suma ponderada de todas estas señales sinápticas. El resultado pasa a la función no lineal de activación que representa el grado de inhibición y/o excitación de la neurona, y una forma de fijar los niveles de actividad de la misma; concluyendo así su modelo estándar (Figura 1).

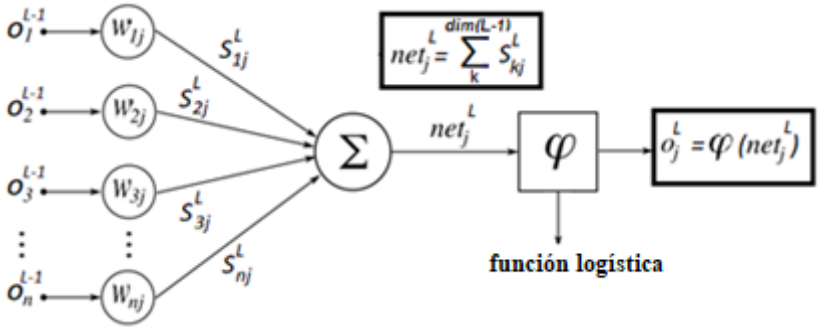


Figura 1. Esquema del modelo estándar de la neurona artificial.

Las redes con capas intermedias reciben el nombre genérico de Perceptrón Multicapa o MLP (Multilayer Perceptron) y son una extensión del perceptrón original con una mayor potencia de procesamiento. Surgieron por las dificultades y limitantes de la arquitectura de perceptrón genérico original donde las posibilidades de obtener una arquitectura de red útil para resolver problemas prácticos eran relativamente pequeñas. Este tipo de red es considerada en la actualidad como un clasificador óptimo y proporciona una gran cantidad de tratamientos para abordar el problema de reconocimiento de patrones [3] [4].

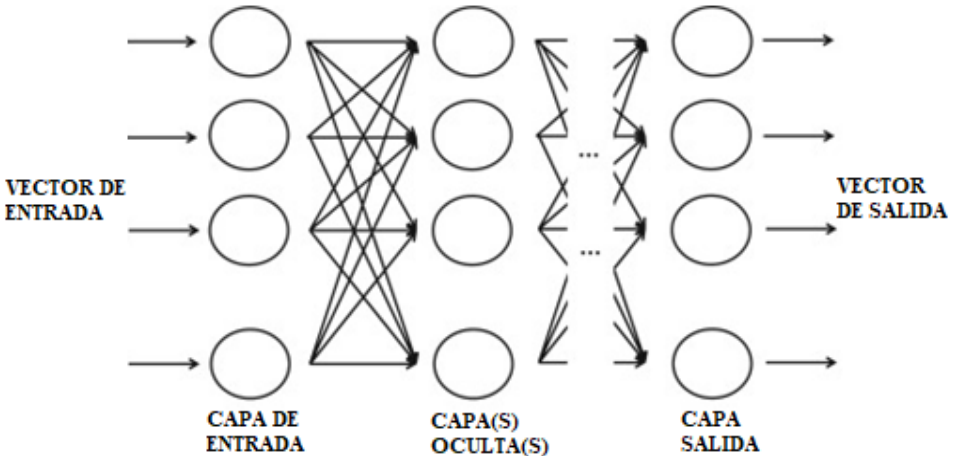


Figura 2. Esquema de una RNA Multicapa (MLP).

Se relaciona con el cerebro en dos aspectos: el conocimiento lo adquiere mediante un proceso de aprendizaje y la fuerza de las conexiones entre neuronas, conocidas como pesos sinápticos, los cuales se emplean para almacenar el conocimiento adquirido [5]. Hipotéticamente, con la red neuronal y pesos apropiados, según el modelo de Rosenblatt, es suficiente para implementar cualquier tipo de tratamiento de la información, pero en la práctica, encontrarlos no es un problema fácil [6].

Muchas investigaciones trabajan con ciertos tipos de redes neuronales que, aunque no son universales, funcionan bien para determinadas tareas. Asociado a cada modelo computacional existe como mínimo un método o algoritmo sistemático para encontrar los valores adecuados de sus pesos. Generalmente se basan en definir una función objetivo para representar de manera global el estado de la red neuronal y, a partir de ella, ir ajustando los pesos establecidos inicialmente a unos valores que la encaminen a un estado estable donde dicha función es mínima.

La adaptabilidad que estos sistemas poseen, es crucial en problemas de reconocimiento; ya que, además de mejorar la tasa de generalización, también permite un buen funcionamiento ante cambios del entorno o presencia de ruido en los datos de entrada. Las RNA pueden beneficiarse de sus resultados en este campo, donde se encuentran actualmente importantes desafíos en el modelado neuronal; pero se encuentran en constante perfeccionamiento, pues en algunos casos se muestran ineficientes.

1.2 PROBLEMAS DEL MODELO DE SINAPSIS ESTÁNDAR

Los pesos sinápticos se han considerado siempre como una forma de ponderar las entradas para dar mayor importancia a unas que otras. Pero en la práctica, el estudio de la relación entre el vector de entrada y el de peso representa el primer paso para la elaboración de un procedimiento de aprendizaje. En el modelo de sinapsis estándar (SS) los vectores de entradas y sus pesos se computan mediante la función de cálculo del valor neto, la cual se expresa como el producto escalar entre los dos vectores (1). Cuanto más alineados se encuentren mayor será su producto, lo cual es una característica acertada a la hora de analizar el grado de alinealidad entre ambos vectores.

$$SS_{ij}^{L} = \left(O_i^{L-1} * W_{ij}^{L}\right) \quad (1)$$

Una problemática a destacar de este modelo y que se intenta resolver en los siguientes apartados, es que presenta un comportamiento indeseado en situaciones donde la magnitud de la entrada es muy grande respecto a su peso, obteniendo un producto alto incluso cuando son muy diferentes (Punto A Figura 3). Este comportamiento conlleva a que la neurona se sature ante la presencia de gran cantidad de señales de entradas con valores altos.

Algo incorrecto también ocurre cuando ambos son iguales, o muy parecidos y pequeños en magnitud (Punto B Figura 3), donde se obtiene un producto pequeño, lo que es inconsistente con la respuesta que se pretende dar al problema.

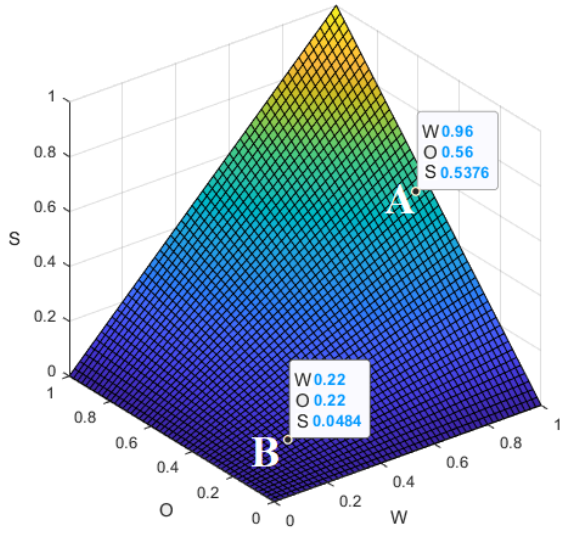


Figura 3. Respuesta del modelo SS para valores de entrada O y peso W en el rango [0, 1].

Como bien se expresa en [7], la saturación ocurre cuando las neuronas de las capas ocultas de una RNA generan predominantemente valores cercanos a los extremos asintomáticos del rango de funciones de activación. La saturación lleva a la red neuronal a un estado binario y limita su capacidad de información general.

En el estudio presentado en [8] se realizó un análisis de las salidas de la función de activación con el objetivo de entender mejor las dificultades que estas manifiestan al entrenar redes neuronales profundas. Actualmente existen investigaciones que demuestran que el aprendizaje por descenso del gradiente puede manifestarse bastante sensible al grado de saturación existente en la red neuronal [9]. Importante destacar que no o existe alguna técnica estandarizada hasta el día de hoy que sea capaz de medir el grado de saturación.

En la siguiente figura se muestra cómo las funciones de activación sigmoidales, que presentan un comportamiento lineal en el rango activo determinado por la pendiente de la función, producen saturación para grandes valores de entrada positivos y negativos.

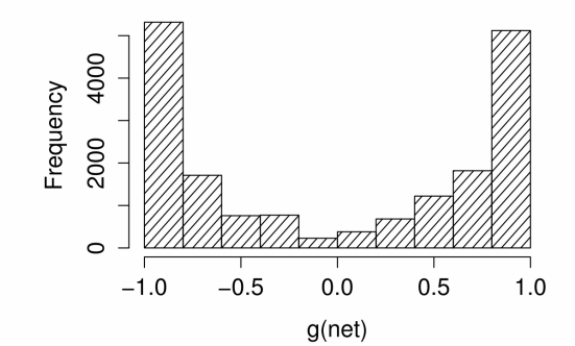


Figura 4. Histograma típico producido por una unidad saturada sobre todos los patrones del conjunto de datos. Cuanto mayor sea la frecuencia en el contenedor más a la izquierda y en el más a la derecha, mayor será la saturación. Tomado de [7].

Si a una neurona que se encuentre saturada se le aplica una pequeña modificación en los pesos de su entrada, no mostrará influencia alguna en su salida. Por lo tanto, un entrenamiento basado en algoritmos de optimización de pesos presentará inconvenientes para determinar si la modificación de este peso tuvo una influencia positiva o negativa en el rendimiento de la red neuronal, provocando que el algoritmo de aprendizaje se estanque.

Uno de los enfoques a desarrollar en este trabajo es intentar corregir esta saturación desde la acción sináptica de la neurona como se plantea en [10], donde implementan un nuevo modelo matemático para la sinapsis que logra eliminar la saturación durante un proceso de reconocimiento con gran cantidad de señales de entrada de alto valor. En una serie de simulaciones logran verificar que el modelo en cuestión presentaba la plasticidad suficiente para incorporar nuevos datos en la neurona.

CAPITULO II. MODELOS DE SINAPSIS SELECTIVAS

En este capítulo pretendemos describir lo más detalladamente posible los nuevos modelos de sinapsis en estudio, con interpretaciones diferentes a la de su homólogo estándar y con características selectivas. Se pretende dar solución a las cuestiones tratadas anteriormente, apoyándonos en casos críticos y representaciones gráficas para su verificación.

2.1 MODELO DE SINAPSIS ABSOLUTA SELECTIVA v1

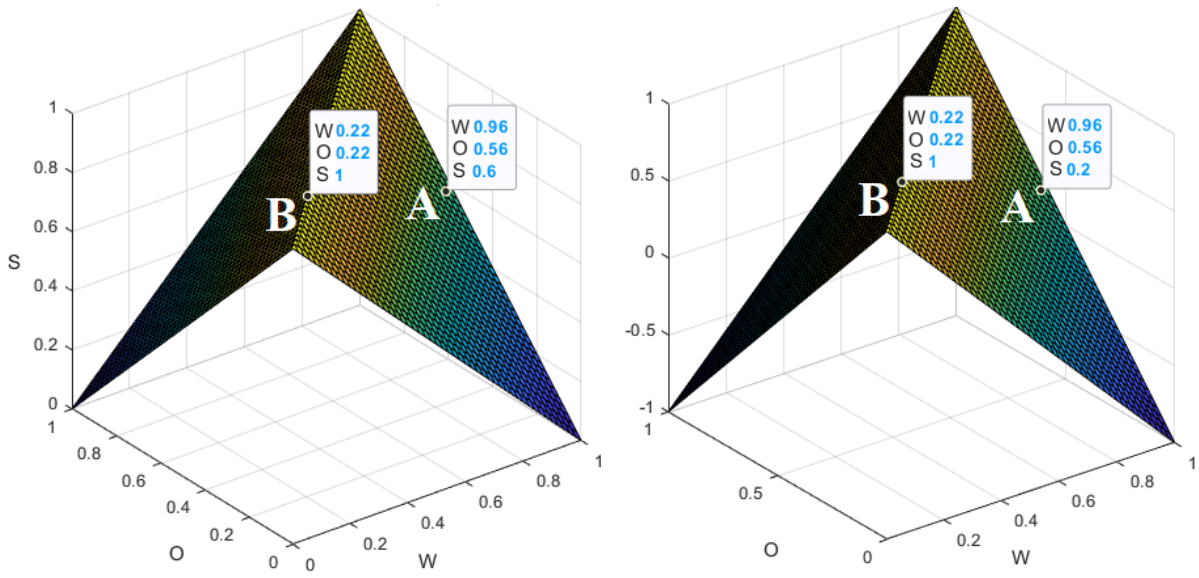
El primer modelo de sinapsis que se propone en este trabajo es el llamado "Sinapsis Absoluta Selectiva v1" (SASv1) y fue planteado para obtener una salida inversamente proporcional a la distancia entre el vector de entrada y su peso (2).

$$SASv1_{ij}^{L} = (1 - \frac{\left|\,|W_{ij}^{L}| - o_{i}^{L-1}\right|}{K}) \tag{2}$$

Las operaciones de cálculo del valor absoluto que la conforman nos garantizan primeramente una resta donde ambos factores son positivos, teniendo en cuenta que las entradas tienen carácter excitatorio; y, por otro lado, nos limita la salida de la neurona con valores máximos y mínimos entre 1 y 0 respectivamente.

Este modelo de sinapsis SASv1 es selectivo comparando el valor a la entrada con su peso. Esta selectividad es proporcionada por el factor K, cuyo valor está en el rango de [0, Inf). Si analizamos la respuesta de esta nueva sinapsis ante los mismos valores usados de ejemplo en la descripción del modelo estándar, realizada en el capítulo anterior; observamos que para el caso A y con factor K=1 (Figura 5 a-)) cuando el peso aleatorio inicial está lejos del valor de la señal de entrada seguimos obteniendo un valor alto a la salida de la sinapsis.

Sin embargo, con factor K=0.5 (Figura 5 b-)) aumentamos el grado de selectividad de la neurona y obtenemos para este caso una salida pequeña. Para el caso B cuando el peso y la señal de entrada tienen valores iguales y pequeños, la nueva sinapsis sí logra generar una señal de salida acorde a lo esperado con su valor máximo para ambos valores del factor K.



a-) con factor K=1 *b-) con factor K=0.5*

Figura 5. Respuesta del modelo SASv1 para valores de entrada O y peso W en el rango [0,1].

2.2 MODELO DE SINAPSIS ABSOLUTA SELECTIVA v2

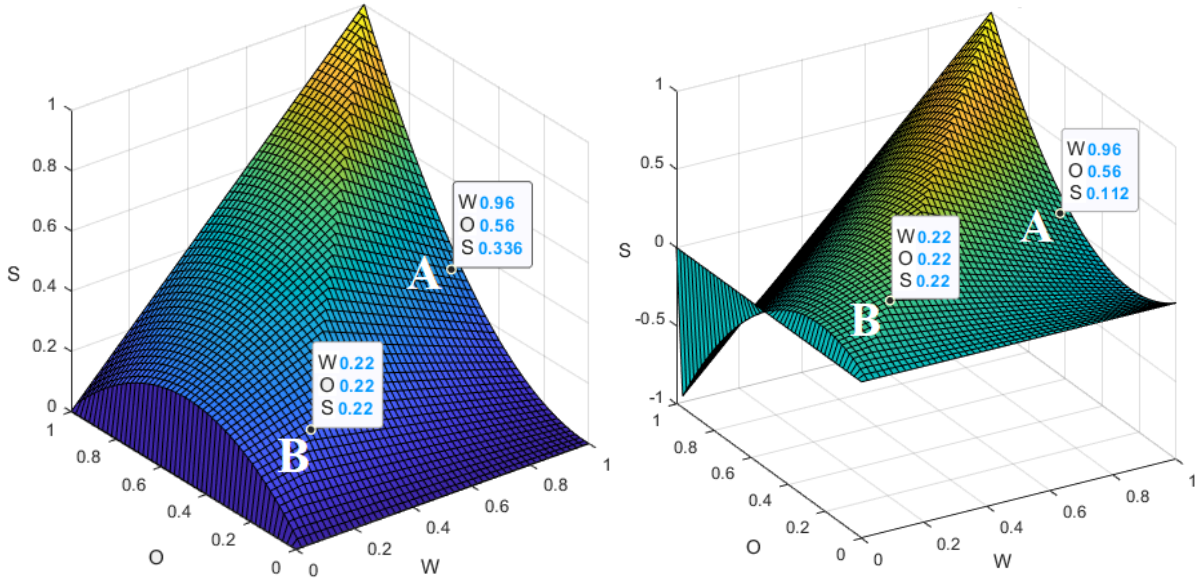
Sobre la base del modelo de sinapsis selectiva anterior, se realizaron modificaciones que llevaron a la obtención de un nuevo modelo de sinapsis (3). Este nuevo planteamiento, al cual llamamos "Sinapsis Absoluta Selectiva v2" (SASv2), tiene incorporado una multiplicación por la señal de entrada y por la función signo de su peso asociado.

$$SASv2_{ij}^{L} = O_i^{L-1} * sign\left(W_{ij}^{L}\right) * \left(1 - \frac{\left|\left|W_{ij}^{L}\right| - o_i^{L-1}\right|}{K}\right) \qquad (3)$$

Con estos cambios se pretende conservar principalmente la ganancia de la señal de entrada que existe en el modelo estándar, permitiendo que su magnitud influya proporcionalmente en la salida de la sinapsis. En cuanto a la función signo, esta fue agregada para recuperar el signo del peso que habíamos perdido en el modelo SASv1 al realizarle la operación del valor absoluto.

El modelo SASv2 intenta mantener algunas características del modelo estándar sin renunciar a las características selectivas aportadas por el factor K y al cálculo del error entre la señal de entrada y su peso, ambos conceptos diferenciadores y novedosos presentes en el modelo SASv1. Como resultado obtenemos una respuesta de este modelo de sinapsis (Figura 6 a-)) con una geometría

similar a la del modelo estándar, pero con una clara discontinuidad para valores de W=0 como consecuencia de la función signo añadida al modelo.



a-) con factor K=1 *b-) con factor K=0.5*

Figura 6. Respuesta del modelo SASv2 para valores de entrada O y peso W en el rango [0,1].

Se aprecia en la Figura 6 b-) cómo al disminuir el valor del factor K aumentamos la selectividad de la sinapsis y con ello el espesor del pico de la gráfica, provocando que, para valores lejanos, como en el caso A, la salida sea más pequeña, y de esta forma disminuye el riesgo de saturación de la neurona. Para el caso B no se obtiene una salida alta, incluso cuando son iguales la entrada y su peso; esto es consecuencia de añadir la ganancia de la señal de entrada a la sinapsis.

2.3 MODELO DE SINAPSIS CUADRÁTICA SELECTIVA

Como solución a las debilidades planteadas anteriormente, se diseñó un nuevo modelo de sinapsis al cual llamamos “Sinapsis Cuadrática Selectiva” (SQS). Este modelo (4) utiliza principalmente el cálculo de la diferencia entre el vector de entrada y su peso, manteniendo el concepto de SASv1, donde esta operación de resta conllevaba a que su salida esté en correspondencia con la similitud de ambos vectores.

$$SQS_{ij}^{L} = 0.1666 - \left(O_i^{L-1} - W_{ij}^{L}\right)^2 / K \qquad (4)$$

Se puede interpretar geométricamente este nuevo modelo como la suma de los cuadrados de la resta de un vector, o sea, el módulo del vector diferencia entre O y W. Como nos interesa que la respuesta de la sinapsis a la salida sea la máxima cuando más se parezcan las componentes y la resta se aproxime a cero, hemos incorporado la sustracción de una constante de valor 0.1666. Este número es el valor medio estadístico de esta diferencia cuadrática y se corresponde con la desviación típica esperada para valores aleatorios con distribución uniforme de W y O (Ver Anexo 4). Su obtención mediante el cálculo del valor medio de una función de dos variables para un intervalo cerrado, se detalla a continuación:

Si W y O son variables aleatorias en el intervalo cerrado [a, b] y [c, d] respectivamente, el valor medio de la función $f(W,O)$ se define como:

$$Fprom = \left(\frac{1}{d-c}\right)\left(\frac{1}{b-a}\right)\int_c^d \int_a^b f(W,O)\, dW\, dO \tag{5}$$

Si $f(W,O) = (W-O)^2$ y los intervalos cerrados para W y O son [0, 1] y [0, 1] respectivamente, obtenemos:

$$Fprom = \left(\frac{1}{1-0}\right)\left(\frac{1}{1-0}\right) \int_0^1 \int_0^1 (W-O)^2 dW\, dO \tag{6}$$

$$= \int_0^1 \left[\left(\frac{W^3}{3} - W^2O + WO^2\right)|_0^1\right] dO$$

$$= \int_0^1 \left[\frac{1}{3} - O + O^2\right] dO \; = \left(\frac{O^3}{3} - \frac{O^2}{2} + \frac{O}{3}\right)|_0^1$$

$$= 0.1666666667$$

En la siguiente figura se aprecia como la señal de salida generada por la sinapsis es la máxima en la diagonal del espacio donde W coincide con O. A medida que W y O difieren, la salida disminuye. Es notable que, para valores de entrada bajos, (Caso A Figura 7) a diferencia del modelo de sinapsis estándar, este nuevo modelo de sinapsis también genera una salida alta. Destacar que el problema de saturación parece resolverse aún en situaciones de abundantes valores altos en las señales de entrada (Caso B Figura 7), donde la sinapsis no llega a saturarse si son suficientemente diferentes.

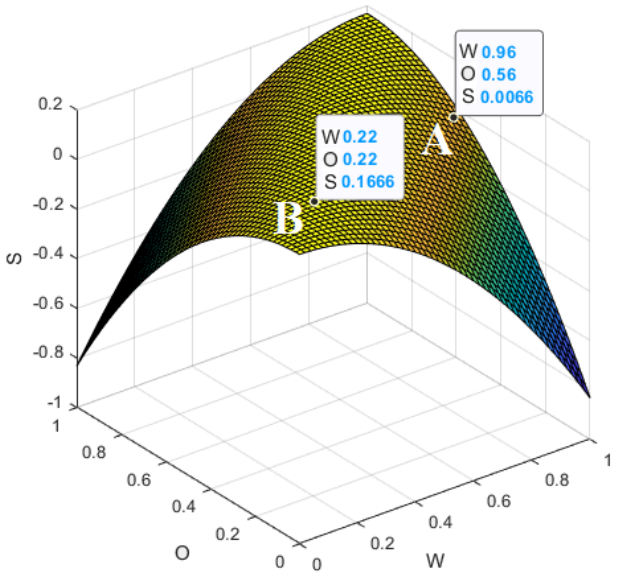


Figura 7. Respuesta del modelo SQS para valores de entrada O y peso W en el rango [0, 1] con factor K=1.

Como nuestra sinapsis es definida positiva, cuanto mayor sea el error, más resto y menor será la salida de la neurona. También está presente el factor K, que de igual forma que los modelos SASv1 y SASv2, determina el grado de selectividad sináptica, dando más o menos importancia a esta comparación. A diferencia del producto escalar utilizado en el modelo convencional de neuronas, este comportamiento nos proporciona una mayor selectividad al comparar las señales de entrada y el peso, lo que es fundamental en un clasificador.

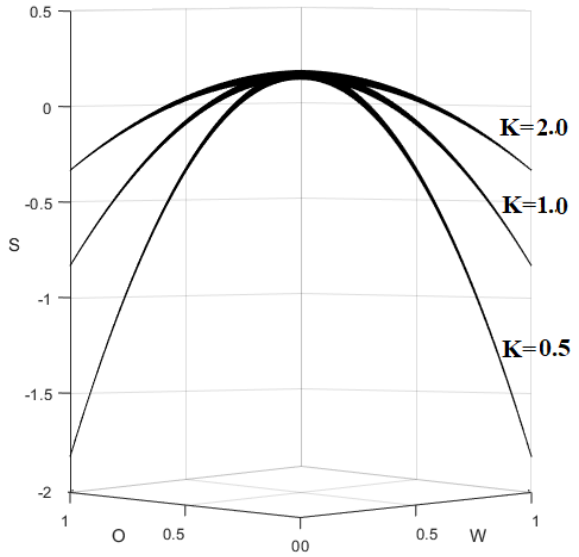


Figura 8. Respuestas del modelo SQS para diferentes valores de factor K.

En la simulación anterior (Figura. 8) se muestran las salidas del modelo SQS para tres valores diferentes del factor K. Se observa como la señal de error de retroalimentación disminuye junto a la selectividad de la sinapsis para K=2>1, obteniéndose una respuesta más amplia. Sin embargo, para K=0.5<1 ocurre lo contrario aumentando el carácter selectivo de la salida de la sinapsis.

CAPITULO III. APRENDIZAJE

En 1974, Paul Werbos desarrolló el primer algoritmo de aprendizaje para redes multicapa [11]. Este algoritmo fue desarrollado para su implementación en diversos tipos de red, con grandes resultados en su aplicación en redes neuronales. Por su capacidad de generalización no fue aceptado por la comunidad de desarrolladores; pero, años después, varios investigadores descubrieron una vez más el llamado “algoritmo de retropropagación”. A raíz de su invención, científicos incluyeron el algoritmo en libros y desarrollaron importantes investigaciones de redes neuronales, alcanzando mucha popularidad y convirtiéndose en una de las redes más utilizadas.

Hoy en día los sistemas computacionales están diseñados para realizar funciones lógicas y matemáticas a una velocidad asombrosa. En cambio, su destreza y capacidad matemática no es una característica necesaria a la hora de resolver el problema de reconocimiento de patrones en un entorno ruidoso, situación que inclusive consumiría mucho tiempo, aunque el espacio de entrada fuese relativamente pequeño. Este problema radica en la naturaleza secuencial de la propia computadora con arquitectura Von Neumann, que le permite a la máquina realizar una única operación en cada ciclo. Por lo que, para aplicaciones que deben explorar una numerosa cantidad de entradas y tratar de correlacionar todas las disposiciones posibles de un conjunto de datos complejos, el tiempo de cálculo requerido se vuelve muy grande.

Uno de los avances significativos del algoritmo de retropropagación fue darle solución a esta problemática con el aprovechamiento de la naturaleza paralela de las redes neuronales, adaptándolas para aprender las relaciones entre conjuntos de patrones dados sin que alguna señal de ruido lo afecte y luego ser capaces de aplicar estas relaciones a datos nuevos. Además, dado que la red puede aprender de manera automática el algoritmo correcto sin tener que desarrollarlo de antemano, se logra reducir significativamente el tiempo de desarrollo de cualquier sistema que se desee analizar.

La retropropagación es una red de aprendizaje supervisado, que utiliza un ciclo adaptativo de propagación en dos etapas. Una vez que el patrón se aplica como estímulo a la entrada de la red, se propagará desde la primera capa a las capas superiores de la red, produciendo así una salida. La señal de salida se compara con la salida deseada y se calcula una señal de error para cada salida.

La señal de error parte de la capa de salida y se propaga hacia las neuronas anteriores en la capa oculta que afectan directamente a esta salida. Estas neuronas en la capa oculta sólo reciben una porción de la señal de error total según su contribución relativa a la salida original. Se repite este proceso en cada una de las capas hasta que todas las neuronas que las conforman reciban una señal de error en correspondencia a su impacto relativo en el error final. En función de la señal de error recibida por cada neurona, esta actualiza su peso de conexión para que la red logre converger a un estado que permita la clasificación correcta de todos los patrones de entrenamiento, de manera más eficiente.

En este proceso de entrenar la red, las neuronas se organizan de tal forma que logran reconocer y responder con una salida activa ante diferentes características del espacio total de entrada que se asemeje a la característica individual que le fue asignada a aprender durante su entrenamiento. De forma contraria, suprime su salida si el patrón a la entrada no la contiene.

3.1 REGLA DE APRENDIZAJE

El entrenamiento de las redes neuronales multicapas se realiza a través de un proceso de aprendizaje donde primeramente debe estar definida la topología de la red, es decir, el número de neuronas en cada capa y el número de capas. La cantidad de neuronas en la capa de entrada está en correspondencia con el número de componentes del vector de entrada y la cantidad de neuronas en la capa de salida depende del tamaño del patrón de destino.

Debe hacerse hincapié en que no existe tecnología que pueda determinar el número de capas ocultas, ni hay una técnica para determinar la cantidad de neuronas que debe contener una red para un problema específico, esta elección depende de la experiencia del diseñador, y los requerimientos de este tipo de cálculo.

Es importante mencionar que no se ha desarrollado hasta nuestros días una técnica que nos permita determinar el número necesario de capas ocultas o el número específico de neuronas que debería contener cada una de estas capas para dar solución a un problema en cuestión; por lo que esta elección cae en manos del diseñador y de su experiencia, estando limitado por las características de tipo computacional para su funcionamiento.

Cada uno de los patrones de entrenamiento se propagan a través de la red y sus parámetros de diseño para generar una respuesta en la salida. Esta salida es comparada con los patrones

objetivo determinando el error en el aprendizaje que marca la ruta más adecuada para la modificación de los pesos. Minimizando el error medio cuadrático en cada iteración se espera que la red genere, al final del aprendizaje, una respuesta acertada ante todos los patrones utilizados en el entrenamiento.

3.2 GENERALIZACION DEL APRENDIZAJE

En este trabajo se desarrollará de forma genérica la deducción matemática de este procedimiento con el objetivo de ser utilizada para el entrenamiento de cada uno de los diseños de red neuronal creados con diferentes modelos de sinapsis.

La función de error total en el proceso de aprendizaje para υ patrones está dada por la ecuación (8).

$$\mathrm{E}_x = \frac{1}{2}\sum_i^{\mathrm{n}} \left(\left(y_x - O_i^p(x) \right)^2 \right) \tag{7}$$

$$\mathrm{E} = \frac{1}{\upsilon}\sum_x^{\upsilon} (\mathrm{E}_x) = \frac{1}{\upsilon}\sum_x^{\upsilon}\sum_i^{n} \frac{1}{2}\left(y_x - O_i^p(x) \right)^2 \tag{8}$$

Donde,

x: vector de entrada (υ vectores de entrada)

y: vector de salidas deseadas (υ vectores de salida asociados a cada entrada)

$O_i^p(x)$*: Salida de la neurona i de la capa p (última capa con n neuronas) cuando se aplica x en las entradas.*

Las funciones de transferencia utilizadas en este tipo de red deben ser continuas para que su derivada exista en todo el intervalo. La función de transferencia utilizada es *logsig* (9) y su respectiva derivada es (9.1).

$$\varphi(x) = \frac{1}{1+e^{-kx}} \tag{9}$$

$$\varphi'(x) = k\frac{e^{-kx}}{\left(1+e^{-kx}\right)^2} = kO(1-O) \tag{9.1}$$

Se recomienda usar k para normalizar en función del número de entradas haciendo k=1/dim(L).

Una red neuronal genera un espacio de k dimensiones con el error producido a su salida en función de sus pesos, donde k es la cantidad de pesos de conexión que contiene dicha red. Para encontrar el conjunto de pesos que minimiza el error es necesario aplicar el método del descenso de gradiente [12] y utilizar la retropropagación para guiar al entrenamiento por la dirección de descenso más pronunciada de una manera eficiente [13].

Este primero implica realizar la derivada parcial del error con respecto a los pesos de la red para así orientarse en dirección negativa del gradiente de la función del error. Para calcular cómo varía el error al variar el peso W_{ij}^{L}, se aplica la regla de la cadena [14], pues el error no es una función explícita de los pesos de la red.

Considerando esto se genera la ecuación:

$$\frac{\partial Ex}{\partial W_{ij}^{L}} = \frac{\partial Ex}{\partial O_{j}^{L}} \frac{\partial O_{j}^{L}}{\partial W_{ij}^{L}} = \frac{\partial Ex}{\partial O_{j}^{L}} \frac{\partial O_{j}^{L}}{\partial net_{j}^{L}} \frac{\partial net_{j}^{L}}{\partial W_{ij}^{L}} \tag{10}$$

Si desarrollamos el tercer término de la ecuación (10):

$$\frac{\partial net_{j}^{L}}{\partial W_{ij}^{L}} = \frac{\partial}{\partial W_{ij}^{L}} \left(\sum_{i}^{\dim (L)} S_{ij}^{L} \right) \tag{11}$$

$$\frac{\partial net_{j}^{L}}{\partial W_{ij}^{L}} = \frac{\partial S_{ij}^{L}}{\partial W_{ij}^{L}} \tag{11.1}$$

Sustituyendo (11.1) en (11) obtenemos:

$$\frac{\partial net_{j}^{L}}{\partial W_{ij}^{L}} = -2\left(O_{i}^{L-1} - W_{ij}^{L}\right)/K \tag{11.2}$$

Ahora desarrollamos el segundo término de la ecuación (10) utilizando (9.1):

$$\frac{\partial O_{j}^{L}}{\partial net_{j}^{L}} = \varphi' = kO_{j}^{L}(1 - O_{j}^{L}) \tag{12}$$

Para desarrollar el primer término de la ecuación (10) debemos tener cuenta dos situaciones, donde la neurona puede pertenecer a la capa de salida o a una capa oculta.

Si la neurona es de salida (capa p), los términos del error para las neuronas de la capa de salida están dados por la ecuación (13.1), la cual se le denomina comúnmente sensitividad de la capa de salida.

$$\frac{\partial Ex}{\partial O_{j}^{p}} = \frac{\partial}{\partial O_{j}^{p}} \left(\sum_{i}^{n} \frac{1}{2} (y_i - O_{i}^{p})^2 \right) \tag{13}$$

$$\frac{\partial Ex}{\partial O_{j}^{p}} = \frac{1}{2} \frac{\partial}{\partial O_{j}^{p}} (y_j - O_{j}^{p})^2 = (O_{j}^{p} - y_j) \tag{13.1}$$

Si la neurona es de una capa oculta (capa *L-1*) definimos:

$$\frac{\partial E_x}{\partial O_j^{L-1}} = \frac{\partial E_x(net_1^L, net_2^L, \ldots, net_{\dim(L)}^L)}{\partial O_j^{L-1}} \tag{14}$$

Para calcular el término de la ecuación (14) se debe aplicar la regla de la cadena en dos ocasiones como se observa en las ecuaciones (14.1) y (14.2).

$$\frac{\partial E_x}{\partial O_j^{L-1}} = \sum_e^{\dim(L)} \left(\frac{\partial E_x}{\partial net_e^L} \; \frac{\partial net_e^L}{\partial O_j^{L-1}}\right) \tag{14.1}$$

$$\frac{\partial E_x}{\partial O_j^{L-1}} = \sum_e^{\dim(L)} \left(\frac{\partial E_x}{\partial O_e^L} \; \frac{\partial O_e^L}{\partial net_e^L} \; \frac{\partial net_e^L}{\partial O_j^{L-1}}\right) \tag{14.2}$$

Si desarrollamos por términos la ecuación (14.2):

$$\frac{\partial net_e^L}{\partial O_j^{L-1}} = \frac{\partial}{\partial O_j^{L-1}}\left(\sum_i^{\dim(L)} S_{ij}^L\right) = \frac{\partial S_{je}^L}{\partial O_j^{L-1}} = doS_{je}^L \tag{14.3}$$

Definimos:

$$\delta_e^L = \frac{\partial E_x}{\partial O_e^L} \; \frac{\partial O_e^L}{\partial net_e^L} \tag{14.4}$$

Sustituimos (14.3) y (14.4) en (14.2):

$$\frac{\partial E_x}{\partial O_j^{L-1}} = \sum_e^{\dim(L)} \delta_e^L \; doS_{je}^L \tag{14.5}$$

Planteamos la derivada de la sinapsis con respecto a la salida:

$$doS_{ij}^L = \frac{\partial S_{ij}^L}{\partial O_i^{L-1}} \tag{15}$$

Si interpretamos (14.4) aplicando (14.5) recursivamente, obtenemos el desarrollo que describe la retropropagación del error. Es la fórmula recursiva en L, calculándose de la salida a la entrada:

$$\delta_e^L = \frac{\partial E_x}{\partial O_e^L} \; \frac{\partial O_e^L}{\partial net_e^L} = \begin{cases} (O_e^L - y_e) kO_e^L(1 - O_e^L), & si\ L\ es\ capa\ de\ salida \\ \left[\sum_k^{\dim(L+1)} (\delta_k^{L+1} \; doS_{ek}^{L+1})\right] kO_e^L(1 - O_e^L), & si\ L\ es\ capa\ oculta \end{cases}$$

Finalmente, vemos cómo hay que modificar el peso W_{ij}^{L}. Para ello vamos a plantear la siguiente expresión sustituyendo (14.4) y (11):

$$\frac{\partial E_x}{\partial w_{ij}^L} = \frac{\partial E_x}{\partial o_j^L}\frac{\partial o_j^L}{\partial net_j^L}\ \frac{\partial net_j^L}{\partial w_{ij}^L} = -\delta_j^L dwS_{ij}^L \tag{16}$$

Luego de encontrar el valor del gradiente del error se procede a actualizar los pesos de todas las capas empezando por la de salida, con la intención de que el error disminuya en cada iteración.

Si empleamos η como velocidad de aprendizaje (η>0), actualizamos los pesos con la siguiente expresión:

$$\Delta W_{ij}^L = -\eta\frac{\partial E_x}{\partial w_{ij}^L} = -\eta\ \delta_j^L dwS_{ij}^L \tag{17}$$

donde, $$\delta_j^L = \begin{cases} (O_j^L - y_j)kO_j^L(1 - O_j^L), & si\ L\ es\ capa\ de\ salida \\ \sum_k^{\dim(L+1)} (\delta_k^{L+1}\ doS_{jk}^{L+1})kO_j^L(1 - O_j^L), & si\ L\ es\ capa\ oculta \end{cases} \tag{17.1}$$

Con el uso de esta técnica de descenso del gradiente es conveniente realizar pequeños incrementos en los pesos para moverse por la superficie de error; debido a que la información que poseemos de la superficie es local y no se conoce cuán lejos o cerca puede estar el punto mínimo. Si establecemos incrementos grandes corremos el riesgo de saltarnos el mínimo; mientras que, con incrementos pequeños, aunque se reduzca la velocidad de convergencia del algoritmo, evitamos que esto ocurra. Por esta razón, se suele elegir un número pequeño para el parámetro η que asegure al algoritmo llegar a una solución, aunque tenga que ejecutar una mayor cantidad de iteraciones.

Este desarrollo matemático del algoritmo Backpropagation es flexible y adaptable para aproximar cualquier función, lo que lo convierte en una de las redes multicapa más potentes; pero su implementación no nos asegura que encontremos el mínimo global, ya que una vez asentado en un mínimo, ya sea local o global, puede quedar fácilmente atascado y culminar el

aprendizaje. En todo caso, conduce a resultados subóptimos que pueden terminar siendo admisible desde el punto de vista del error, aunque nos encontremos en un mínimo local.

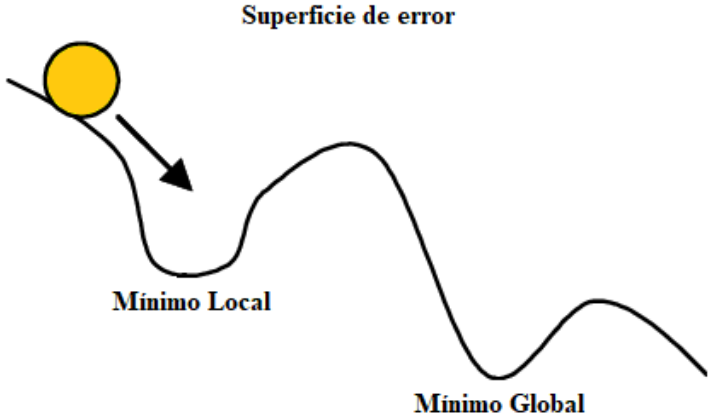


Figura 9. Representación de la superficie de error con mínimos locales y globales.

Para evitar esta situación, utilizamos un término de impulso en la función objetivo, que es un valor entre 0 y 1 que aumenta el tamaño de los pasos dados hacia el mínimo al tratar de saltar de un mínimo local. Si el término de impulso es grande, entonces la tasa de aprendizaje debe mantenerse más pequeña. Este impulso se implementa añadiendo en el aprendizaje un momento de inercia, donde en el cambio actual influye una fracción del anterior, lo que mejora la convergencia evitando oscilaciones. Si se añade una inercia en el cambio:

$$\Delta W_{ij}^{L}(t) = \eta\ \delta_j^L dS_{ij}^L + \xi\ \Delta W_{ij}(t-1) \tag{18}$$

Luego de haber desarrollado la generalización del algoritmo de aprendizaje debemos plantear las derivadas de cada modelo de sinapsis en particular y que debemos sustituir en las ecuaciones (11.1) y (15) según el modelo a entrenar. La primera de estas ecuaciones describe la derivada del modelo de sinapsis respecto al peso ($\frac{\partial s_{ij}^L}{\partial w_{ij}^L}$) mientras que la otra ecuación describe la derivada del modelo respecto a su entrada ($\frac{\partial s_{ij}^L}{\partial o_i^{L-1}}$). Todos estos desarrollos se muestran en la siguiente tabla.

Tabla 1. Desarrollo de las derivadas para cada modelo de sinapsis a entrenar.

Modelo de Sinapsis	Derivadas	
SASv1	$\frac{\partial s_{ij}^L}{\partial w_{ij}^L} = W_{ij}^L \frac{O_i^{L-1} - \lvert w_{ij}^L \rvert}{K * \lvert w_{ij}^L \rvert * \lvert w_{ij}^L \rvert - O_i^{L-1} \rvert}$	(19)
	$\frac{\partial s_{ij}^L}{\partial O_i^{L-1}} = \frac{\lvert w_{ij}^L \rvert - O_i^{L-1}}{K * \lvert w_{ij}^L \rvert - O_i^{L-1} \rvert}$	(20)
SASv2	$\frac{\partial s_{ij}^L}{\partial w_{ij}^L} = O_i^{L-1} \left(\frac{-sign\left(\lvert w_{ij}^L \rvert - O_i^{L-1} \right)}{K} \right)$	(21)
	$\frac{\partial s_{ij}^L}{\partial O_i^{L-1}} = sign\left(W_{ij}^L \right) \left(\frac{1 - \left\lvert \lvert w_{ij}^L \rvert - O_i^{L-1} \right\rvert}{K} \right) + O_i^{L-1} sign\left(W_{ij}^L \right) \frac{sign\left(\lvert w_{ij}^L \rvert - O_i^{L-1} \right)}{K}$	(22)
SQS	$\frac{\partial s_{ij}^L}{\partial w_{ij}^L} = 2(O_i^{L-1} - W_{ij}^L)/K$	(23)
	$\frac{\partial s_{ij}^L}{\partial O_i^{L-1}} = -2(O_i^{L-1} - W_{ij}^L)/K$	(24)

3.3 BASE DE DATOS

El algoritmo de retropropagación para una red multicapa es una generalización del algoritmo Learning Management System (LMS) ya que ambos se basan en el error cuadrático medio para realizar su función de actualizar los pesos y ganancias de la red. La red de retropropagación funciona bajo aprendizaje supervisado como mencionábamos anteriormente, por lo que es necesario la existencia de un conjunto de datos para el entrenamiento que contenga las entradas y sus valores de salida deseados.

El conjunto de datos que usaremos en este trabajo se denomina conjunto de datos MNIST [15] y es uno de los más utilizados en la comunidad de aprendizaje automático. Es un subconjunto de un conjunto más grande disponible en NIST, con 60.000 patrones para entrenamiento y 10.000 patrones para validación de aproximadamente 500 escritores. Este conjunto de datos está conformado por imágenes de dígitos escritos a mano ajustados en cajas de tamaño fijo 20x20 píxeles, normalizadas y centradas.

El motivo y ventaja de su utilización está en la reducción de tiempo y esfuerzo en preprocesamiento y formato que nos tomaría su creación. Algunos ejemplos de los dígitos incluidos en este conjunto de datos se muestran en la siguiente figura.

Figura 10. Ejemplos de patrones incluidos en el conjunto de datos MNIST de dígitos escritos a mano.

3.4 ARQUITECTURA DE LAS REDES

Las arquitecturas de todas las rede neuronales simuladas en este trabajo van a definirse de igual forma, con el objetivo de realizar una posterior comparación donde no influyan estos factores. Diferentes arquitecturas pueden producir resultados dramáticamente diferentes, ya que el rendimiento se puede considerar como una función de la arquitectura, entre otras cosas, como los parámetros, los datos y la duración del entrenamiento. Todas las redes se conformarán por tres capas y 35 neuronas; distribuyéndose en 16 neuronas en la capa de entrada, 15 en la capa oculta y 4 en la capa de salida. Se han diseñado las redes con una capa oculta para que logren aprender entidades en varios niveles de abstracción, y de esta forma sean más capaces de generalizar [16].

Para representar las imágenes en la entrada, los píxeles de 20x20 que la componen se aplanan en un vector de 1D de 400 píxeles de tamaño, donde cada uno se almacena como un valor entre 0 y 1. Por otro lado, para leer a la salida de las redes neuronales, estamos utilizando una codificación en binario de 4 bits para hacer corresponder con las etiquetas del dígito real dibujado. El vector de salida nos otorga 16 posibilidades, pero sólo utilizamos las 10 correspondientes al rango de dígitos del "0" hasta la "9", uno para cada dígito posible. Se

estableció el límite de error de la codificación a la salida en 0.015; o sea, si el error total entre todas las componentes del vector es menor que este valor podemos afirmar un correcto reconocimiento del dígito, en caso contrario para valores mayores de error se considera fallido. El siguiente diagrama muestra una visualización de la arquitectura que hemos diseñado, con cada capa totalmente conectada a las capas circundantes:

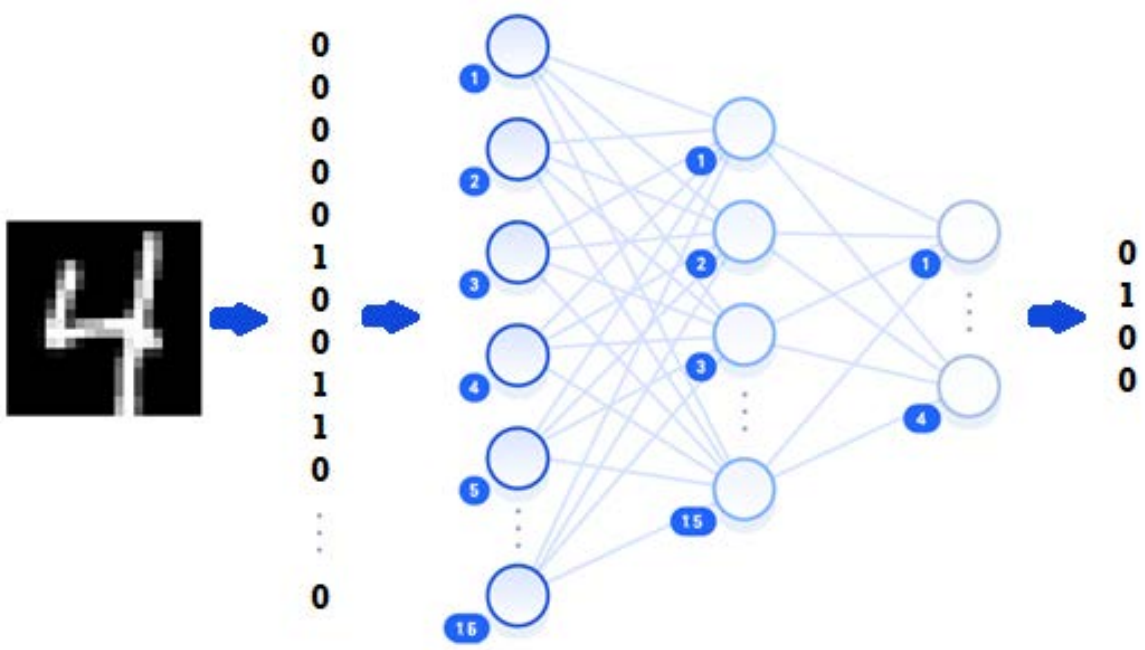


Figura 11. Diagrama de la arquitectura diseñada para las redes neuronales utilizadas.

CAPITULO IV. SIMULACIONES

En la aplicación de las redes neuronales artificiales para la resolución de problemas podemos encontrar relaciones muy complejas entre los parámetros de su diseño, lo que dificulta la búsqueda de los valores que optimicen el proceso; pues provoca que se convierta en una actividad extremadamente lenta el llevar a cabo todos los experimentos combinatoriamente posibles con el fin de comprender estas relaciones.

Cada cambio de valor de cualquier parámetro de la red neuronal requiere un nuevo análisis de la misma, y en correspondencia para cada situación se genera una superficie de respuesta en función de todas estas variables. Por esta razón es necesario identificar en esta etapa del proceso los parámetros de la arquitectura que pueden afectar más al rendimiento y analizarlos; teniendo en cuenta que el objetivo que se quiere alcanzar es disminuir el error entre la salida de la RNA y la deseada para cada patrón de entrada y consecuentemente lograr una mayor tasa de reconocimiento de todos los patrones de la base de datos.

4.1 APRENDIZAJE

Los hiperparámetros de una RNA dan forma a su funcionamiento determinando su precisión y validez. A diferencia de los parámetros que se actualizarán durante el entrenamiento, estos valores se establecen inicialmente y permanecen constantes durante todo el proceso. El ajuste de hiperparámetros siempre se realiza con una métrica o puntuación de optimización, mediante la cual se intenta determinar los valores óptimos para diferentes valores de hiperparámetros. La métrica de optimización utilizada en este trabajo para medir el desempeño de cada red neuronal es la precisión en el reconocimiento o tasa de aciertos (25).

$$Tasa\ de\ Aciertos = \frac{patrones\ reconocidos}{Total\ de\ patrones} * 100 \quad (\%) \qquad (25)$$

Si se optimiza a ciegas la precisión y se ignora el sobreajuste, se obtiene como resultado un modelo muy preciso para el conjunto de entrenamiento, pero que no funcionará bien con muestras desconocidas. La validación ayuda a garantizar que no está optimizando la precisión a expensas del ajuste del modelo. Por lo tanto, fue dividida la base de datos MNIST

en dos subconjuntos. La cantidad de patrones para el entrenamiento determinan la cantidad de características a aprender por la red, siempre y cuando la red posea la capacidad y tamaño para identificar todas estas relaciones existentes. Se debe tener en cuenta que mientras mayor sea este conjunto de entrenamiento más demorará el aprendizaje, algo necesariamente importante a considerar pues se simularán gran cantidad de experimentos. Además, como sólo se pretende analizar el comportamiento de los hiperparámetros de cada red y comparar sus resultados ante iguales circunstancias, se entrenarán con sólo 200 patrones y, posteriormente, se ejecutará con los restantes 9800 patrones de validación.

El número de veces que el algoritmo entrena todo el conjunto de patrones se denomina "épocas". En las simulaciones siguientes este parámetro fue fijado a 1000 y se impuso como una de las condiciones para culminar el entrenamiento junto con el error de convergencia con valor 0.25; este último representa el error acumulado en el ejercicio de reconocimiento de los patrones del conjunto de entrenamiento.

Las opciones básicas para la inicialización de pesos son establecer los pesos en cero o aleatorizarlos. Sin embargo, esto puede resultar en un gradiente que desaparece o influenciar la convergencia del algoritmo que dificulta el entrenamiento del modelo. Para mitigar este problema y realizar un análisis razonable se van a utilizar los mismos pesos iniciales para cada red determinados de forma aleatoria en el intervalo [0, 0.2], garantizando que estén cerca de cero para un mejor ajuste y de valores positivos para que la comparación realizada en los nuevos modelos de sinapsis sea más consecuente, teniendo en cuenta que las señales de entrada también lo son.

Para medir el comportamiento de las redes ante la variación de cada uno de los hiperparámetros se pretende realizar una búsqueda sistemática que implica probar varios valores de los mismos, reentrenando automáticamente el modelo para cada valor del parámetro. De esta forma podemos realizar un estudio de su influencia y del espacio problemático, proporcionándonos más claridad para conseguir la optimización. Teniendo en cuenta que puede ser lento en ejecutarse para un gran número de valores de hiperparámetros se seleccionaron 10 valores consecutivos de un intervalo establecido, para realizar el análisis de cada parámetro en particular.

Los hiperparámetros interactúan entre ellos, pero para fines prácticos se pueden considerar sus ajustes de manera independiente, ya que estas interacciones no tienen una estructura

aparente. Por esta razón, el orden a seguir en el ajuste de los hiperparámetros es en gran medida subjetivo, aunque en investigaciones realizadas recomiendan ajustar la tasa de aprendizaje primero [17]. Siguiendo esta recomendación se estudió la influencia de los hiperparámetros tasa de aprendizaje, ganancia de la neurona y momento de inercia, siguiendo este orden, para cada red neuronal con su modelo de sinapsis.

4.1.1 APRENDIZAJE CON MODELO DE SINAPSIS ESTÁNDAR

Después de conformada la red con el modelo SS se pasó a la ejecución de los 30 experimentos correspondientes, 10 experimentos para cada hiperparámetro. Los resultados almacenados nos proporcionan el tiempo empleado para el aprendizaje, la tasa de aciertos con el conjunto de entrenamiento y la tasa de acierto con el conjunto de validación, para cada valor de hiperparámetro seleccionado como se muestra en la Tabla 2. Los resultados para cada valor de hiperparámetro fueron graficados para observar mejor su comportamiento teniendo en cuenta estos factores.

Tabla 2. Resultados del entrenamiento de la RNA con modelo SS para cada uno de los valores de hiperparámetros utilizados.

Tasa de Aprendizaje	**0,001**	**0,003**	**0,007**	**0,01**	**0,03**	**0,07**	**0,1**	**0,3**	**0,7**	**1**
Tiempo Aprendizaje	19,88	19,38	17,95	18,52	21,23	19,47	18	13,33	9,7	8,25
Tasa de Acierto CE	100	100	100	100	100	100	100	100	100	100
Tasa de Acierto CV	55,55	54,28	58,09	55,67	55,75	55,38	56,04	54,43	60,93	60,53
Ganancia Neurona	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
Tiempo Aprendizaje	39,89	18,44	13,2	11,85	9,03	22,51	139,84	137,83	138,79	158,16
Tasa de Acierto CE	100	100	100	100	100	100	0	0	22	14
Tasa de Acierto CV	56,91	57,63	56,22	61,74	62,65	59,14	0	0	15,67	11,29
Momento de Inercia	**0,1**	**0,2**	**0,3**	**0,4**	**0,5**	**0,6**	**0,7**	**0,8**	**0,9**	**1**
Tiempo Aprendizaje	19,4	12,58	9,69	14,92	29,77	135	136,16	137,35	138,85	138,64
Tasa de Acierto CE	100	100	100	99,5	99,5	22,5	10	0	0	14
Tasa de Acierto CV	51,56	57,21	61,43	57,94	59,11	16,84	8,89	0	0	9,73

Como se ha expresado anteriormente y se corresponde con lo observado en la Figura 12, la tasa de aprendizaje influye en la convergencia del algoritmo. Si el valor de la tasa de aprendizaje es demasiado pequeño, los pasos son pequeños y la optimización es lenta. Para este caso, la curvatura de la función de costo parece ser baja, por lo que resulta mejor una mayor tasa de aprendizaje. Se debe tener en cuenta que, para valores demasiado grandes, los pasos serían extensos y aumentaría la probabilidad de que la optimización diverja.

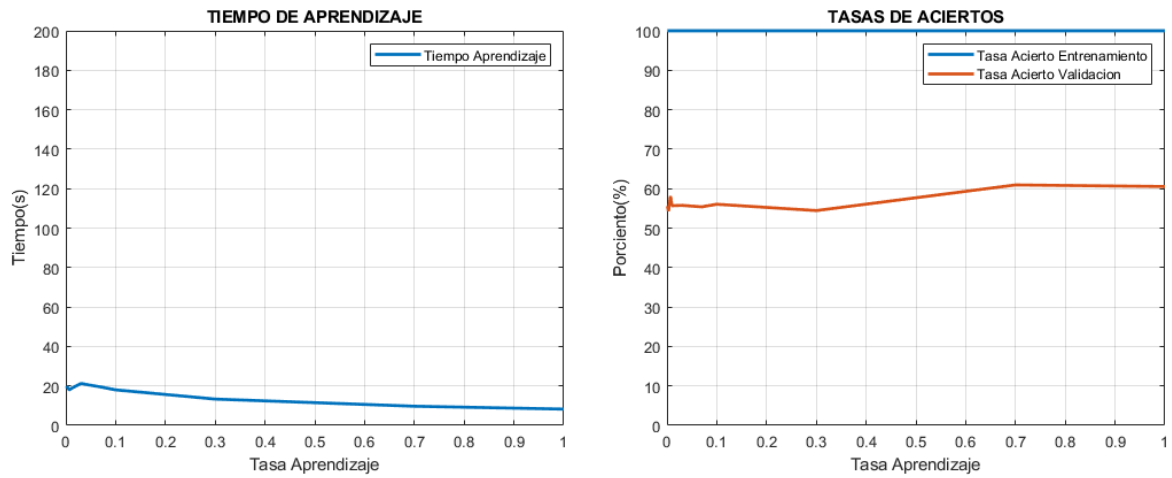


Figura 12. Comportamiento de la RNA con modelo SS para diferentes valores de tasa de aprendizaje.

En el caso de la ganancia de la neurona, se intenta determinar la ganancia óptima de la actividad de la neurona para una mejor precisión final. Su aumento o disminución no influye en la fuerza de la sinapsis, pero un nuevo modelo de sinapsis sí conlleva a un nuevo ajuste del mismo. En una RNA con modelo SS, valores grandes de ganancia provocan la saturación de neuronas y con ello aportes erróneos (Figura 13).

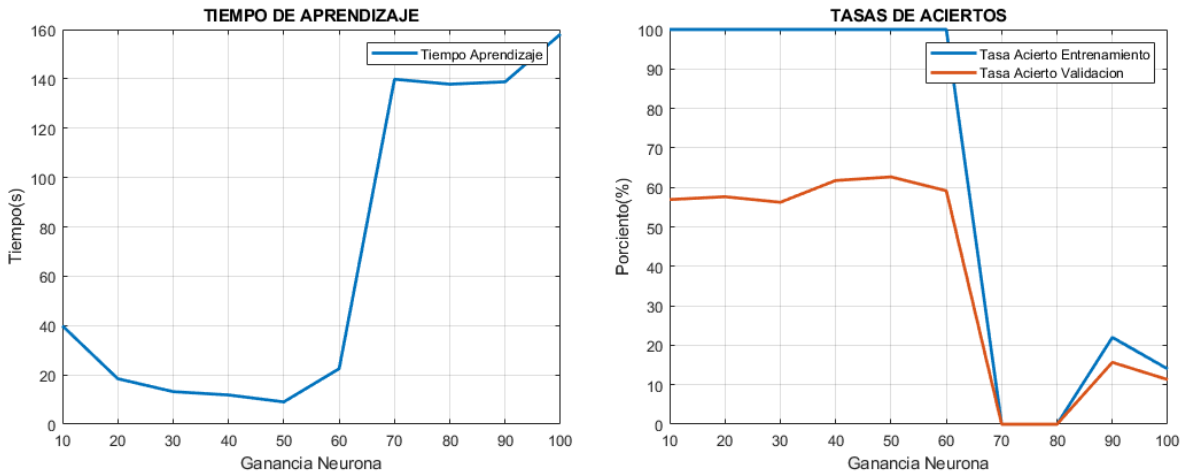


Figura 13. Comportamiento de la RNA con modelo SS para diferentes valores de ganancia de la neurona.

Este proceso, donde los pesos se actualizan para que puedan converger hacia el mínimo de la función de pérdida, puede resultar demasiado extenso y afectar a la eficiencia del algoritmo. El momento de inercia es una posible solución, intentando mantener un seguimiento de las direcciones anteriores como información incrustada. Como se observa en la Figura 14, su valor óptimo resulta en un aumento de la velocidad de convergencia.

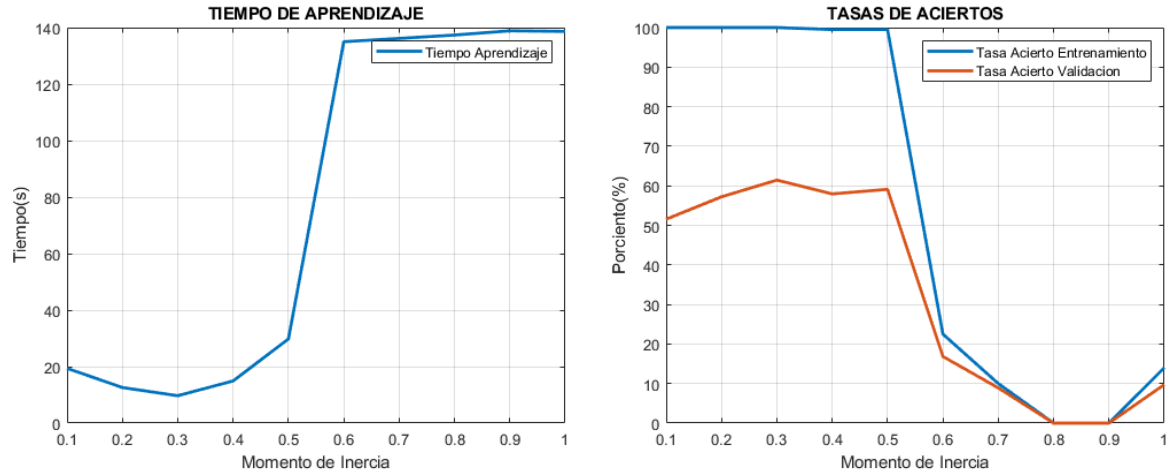


Figura 14. Comportamiento de la RNA con modelo SS para diferentes valores de momento de inercia.

4.1.2 APRENDIZAJE CON MODELO DE SINAPSIS ABSOLUTA SELECTIVA v1

Siguiendo el mismo procedimiento, se realizaron los experimentos para la RNA con el modelo SASv1, pero no se obtuvieron buenos resultados en el aprendizaje. Cuando se realiza el entrenamiento la red no muestra disminución en el error (Figura 15) y no muestra signos de aprendizaje para los patrones de entrenamiento.

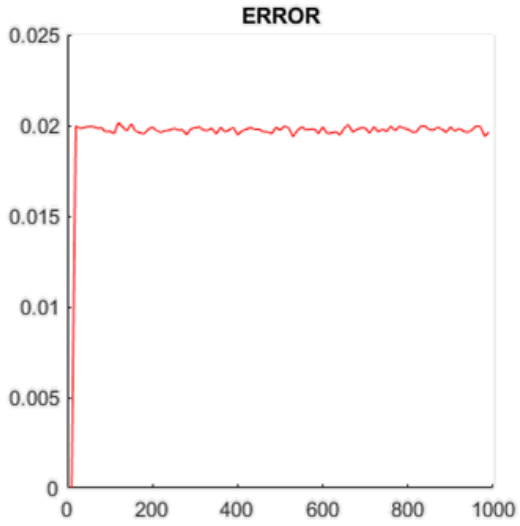


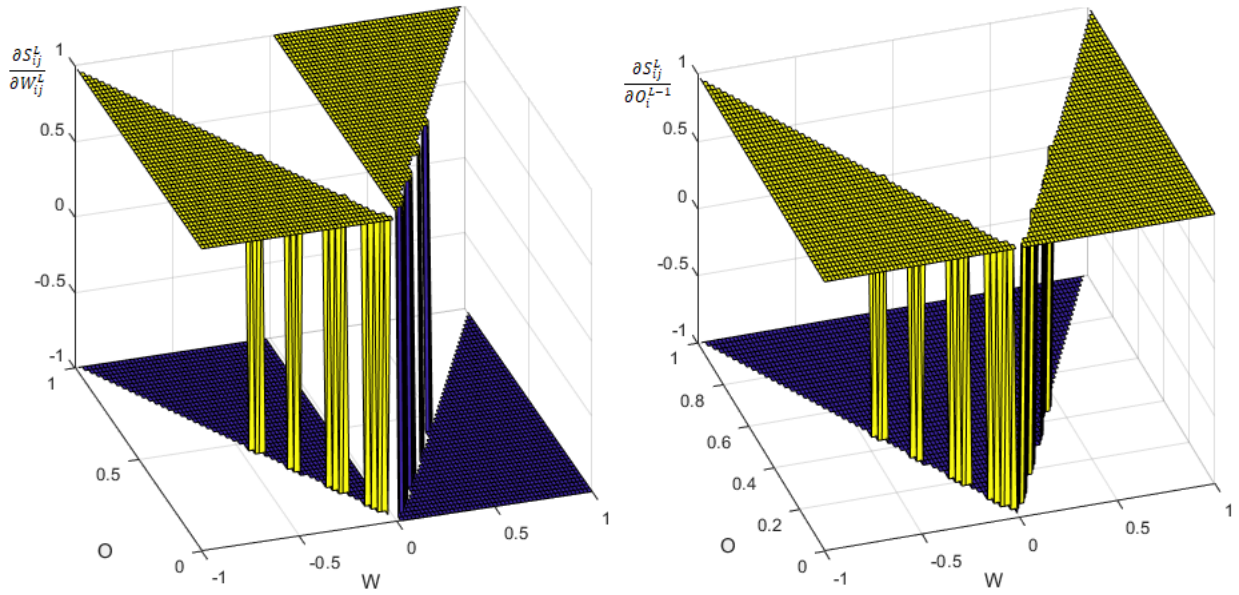
Figura 15. Gráfica del error durante el entrenamiento de la RNA con modelo SASv1.

Se verificó que el algoritmo funcionaba correctamente y que se realizaban actualizaciones a los pesos sinápticos; sin embargo, estas actualizaciones no fueron correspondientes con el aporte particular de cada neurona al reconocimiento, o sea, durante el entrenamiento con la retropropagación del error, las derivadas de la sinapsis respecto al peso W_{ij}^L (26) y respecto a la entrada O_i^{L-1} (27) no contenían la ganancia de la entrada O_i^{L-1} y, por tanto, no contenían su contribución relativa. Las actualizaciones de los pesos se realizaban en el sentido correcto, pero con magnitudes que sólo dependían del factor K como se muestra en el desarrollo realizado a continuación.

$$\frac{\partial s_{ij}^L}{\partial w_{ij}^L} = W_{ij}^L * \frac{o_i^{L-1} - |w_{ij}^L|}{K * |w_{ij}^L| * \left| |w_{ij}^L| - o_i^{L-1} \right|} = \begin{cases} -\frac{1}{K}, & W_{ij}^L > 0 \;\; y \;\; (|W_{ij}^L| - O_i^{L-1}) > 0 \\ -\frac{1}{K}, & W_{ij}^L < 0 \;\; y \;\; (|W_{ij}^L| - O_i^{L-1}) < 0 \\ +\frac{1}{K}, & W_{ij}^L < 0 \;\; y \;\; (|W_{ij}^L| - O_i^{L-1}) > 0 \\ +\frac{1}{K}, & W_{ij}^L > 0 \;\; y \;\; (|W_{ij}^L| - O_i^{L-1}) < 0 \end{cases} \tag{26}$$

$$\frac{\partial s_{ij}^{L}}{\partial o_{i}^{L-1}} = \frac{|w_{ij}^{L}|-o_{i}^{L-1}}{K*||w_{ij}^{L}|-o_{i}^{L-1}|} = \begin{cases} -\frac{1}{K}, & (|W_{ij}^{L}| - O_{i}^{L-1}) < 0 \\ +\frac{1}{K}, & (|W_{ij}^{L}| - O_{i}^{L-1}) > 0 \end{cases} \quad (27)$$

Si se observa, las siguientes figuras también corroboran la existencia de gran cantidad de discontinuidades en las derivadas, lo que dificulta el proceso de aprendizaje cuando las variables obtienen determinados valores críticos.



a-) *b-)*

Figura 16. Respuesta de las derivadas de la sinapsis SASv1 a-) respecto al peso W y b-) respecto a la entrada O, para un rango de valores de los mismos y con factor K=1.

4.1.3 APRENDIZAJE CON MODELO DE SINAPSIS ABSOLUTA SELECTIVA v2

Se realizaron las simulaciones para la RNA con el modelo SASv2. En este caso el error que producía el no-aprendizaje de la red con modelo SASv1 fue corregido al añadirse la multiplicación de la ganancia de la entrada como se detalla en la descripción del modelo en el anterior capítulo. Los resultados para cada valor de hiperparámetro se muestran en la siguiente Tabla 3.

Tabla 3. Resultados del entrenamiento de la RNA con modelo SASv2 para cada uno de los valores de hiperparámetros utilizados.

Tasa de Aprendizaje	**0,001**	**0,003**	**0,007**	**0,01**	**0,03**	**0,07**	**0,1**	**0,3**	**0,7**	**1**
Tiempo Aprendizaje	174,7	180,5	184,5	186,0	182,8	183,7	208,9	192,7	76,83	30,59
Tasa de Acierto CE	100	100	100	100	99,5	100	100	100	100	99,5
Tasa de Acierto CV	50,17	52,34	52,14	48,95	50,58	50,64	49,74	51,26	54,53	53,82
Ganancia Neurona	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
Tiempo Aprendizaje	166,3	18,33	16,43	39,34	39,83	184,7	182,3	191,0	201,3	234,4
Tasa de Acierto CE	100	100	100	100	100	62	71	19	8,5	28
Tasa de Acierto CV	51,21	53,52	57,36	57,52	57,01	39,46	37,47	17,74	9,82	17,72
Momento de Inercia	**0,1**	**0,2**	**0,3**	**0,4**	**0,5**	**0,6**	**0,7**	**0,8**	**0,9**	**1**
Tiempo Aprendizaje	64,98	213,24	136,9	44,74	58,25	194,95	88,7	91,81	26,96	41,06
Tasa de Acierto CE	100	14	100	100	100	24	100	100	100	100
Tasa de Acierto CV	58,25	9,73	52,74	50,85	56,07	16,47	51,4	56,79	54,32	54,48

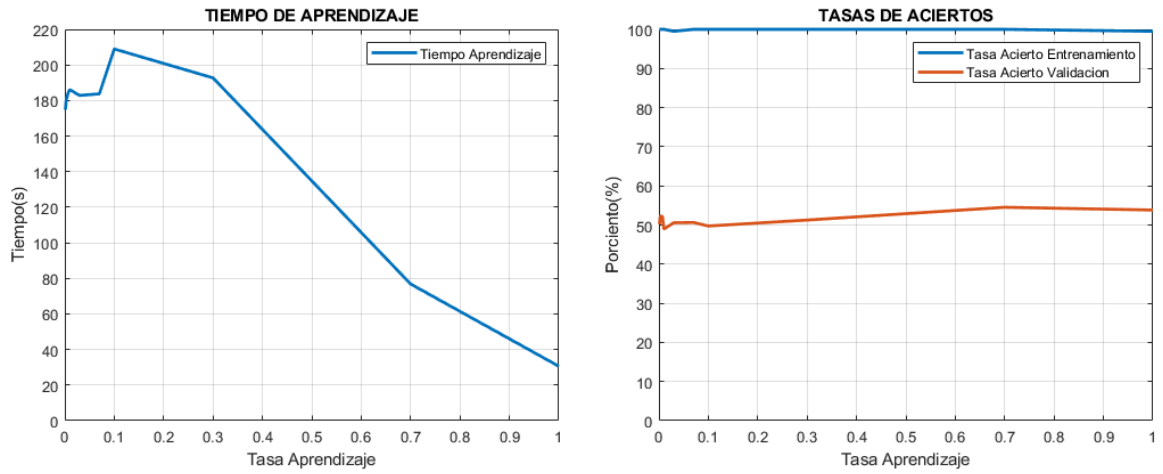


Figura 17. Comportamiento de la RNA con modelo SASv2 para diferentes valores de tasa de aprendizaje.

Como se observa en la Figura 17, la red con el nuevo modelo SASv2 presenta, al igual que la red con modelo SS, un mejor rendimiento para valores grandes de la tasa de aprendizaje, pero demoras mucho

mayores en el entrenamiento a medida que disminuimos su valor. Destacar que esta red presenta en este intervalo gran estabilidad en el porcentaje de reconocimiento.

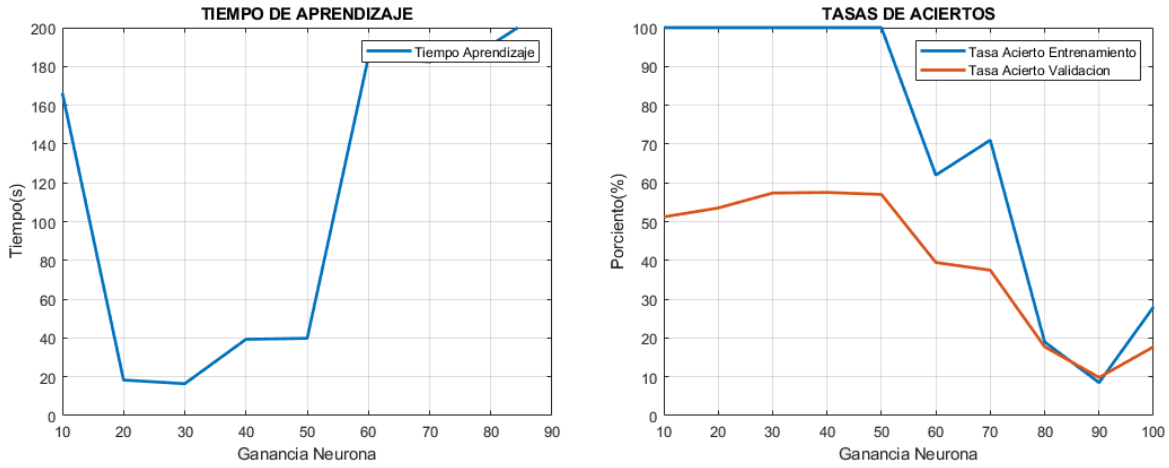


Figura 18. Comportamiento de la RNA con modelo SASv2 para diferentes valores de ganancia de la neurona.

Para la elección de los valores de ganancia óptima de la neurona se debe tener en cuenta que para la red con este modelo SASv2, los valores altos suelen empeorar el aprendizaje; este comportamiento es similar al de la red con modelo SS, lo que resulta completamente coherente por las características de este modelo que le fueron incorporadas al modelo SASv2.

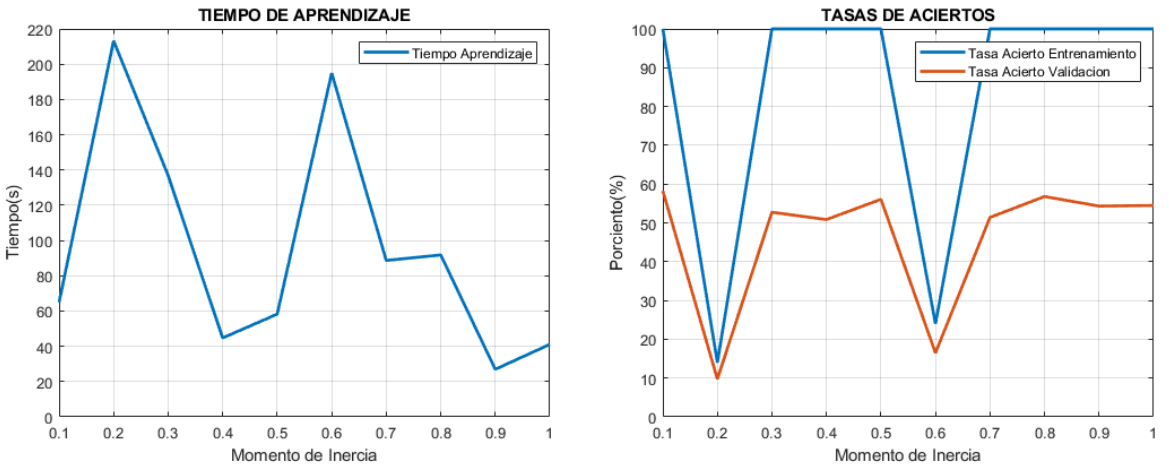


Figura 19. Comportamiento de la RNA con modelo SASv2 para diferentes valores de momento de inercia.

Por otro lado, en las gráficas del hiperparámetro momento de inercia se visualizan rangos de valores bastante dispersos con buenos resultados (Figura 19). La red manifiesta un comportamiento inestable en este intervalo, pero a su vez logra, con algunos valores, realizar un posterior reconocimiento con porcentajes de aciertos competitivos con respecto a la red con modelo SS. Es un campo sin características deducibles a simple vista y sin una clara metodología a seguir para su optimización.

4.1.4 APRENDIZAJE CON MODELO DE SINAPSIS CUADRATICA SELECTIVA

La última RNA a analizar con el modelo SQS fue conformada y simulada manteniendo el mismo procedimiento, pero modificando los intervalos de los hiperparámetros de tasa de aprendizaje y momento de inercia debido a que no presentaba un buen funcionamiento para los valores de los experimentos anteriores. Los resultados para cada valor de hiperparámetro utilizado se muestran en la Tabla 4.

Tabla 4. Resultados del entrenamiento de la RNA con modelo SQS para cada uno de los valores de hiperparámetros utilizados.

Tasa de Aprendizaje	**0,0001**	**0,0003**	**0,0007**	**0,001**	**0,003**	**0,007**	**0,01**	**0,03**	**0,07**	**0,1**
Tiempo Aprendizaje	49,57	55,78	32,03	40,43	17,2	8,11	6,29	4,02	174,1	181,4
Tasa de Acierto CE	100	100	100	100	100	100	100	100	0	14
Tasa de Acierto CV	57,75	57,8	57,97	57,83	57,85	57,81	57,69	57,71	0	11,29
Ganancia Neurona	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
Tiempo Aprendizaje	163,8	162,8	165,6	163,8	164,2	163,9	105,7	70,05	58,2	65,58
Tasa de Acierto CE	0	43	95	100	100	100	100	100	100	100
Tasa de Acierto CV	0	24,25	45,6	51,1	50,3	50,03	51,02	48,39	46,04	46,38
Momento de Inercia	**0,0001**	**0,0003**	**0,0007**	**0,001**	**0,003**	**0,007**	**0,01**	**0,03**	**0,07**	**0,1**
Tiempo Aprendizaje	83,1	75,83	74,43	62,74	46,62	29,89	18,83	8,59	162,09	171,79
Tasa de Acierto CE	100	100	100	100	100	100	100	100	0	8,5
Tasa de Acierto CV	44,74	47,75	48,31	47,87	47,96	47,89	48,06	50,62	0	9,82

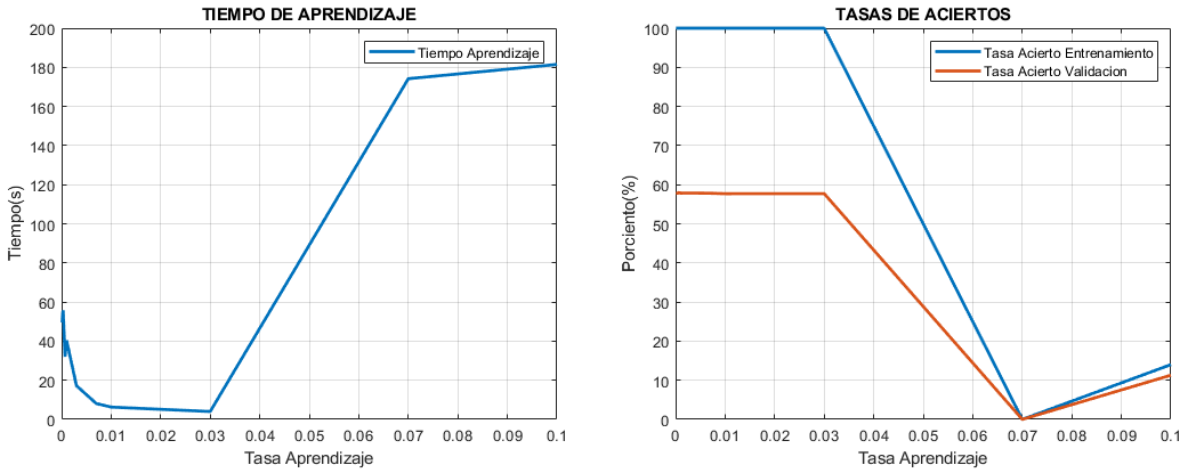


Figura 20. Comportamiento de la RNA con modelo SQS para diferentes valores de tasa de aprendizaje.

Es apreciable un comportamiento más suave en esta red con modelo SQS, alcanzándose los mejores resultados con tasas de aprendizaje pequeñas a diferencia de las redes con modelo SS y SASv2. La metodología a seguir para su ajuste en este tipo de red con modelo SQS sería inversa a la del modelo SS, empezando por un valor más alto e ir disminuyéndolo hasta alcanzar su rendimiento óptimo.

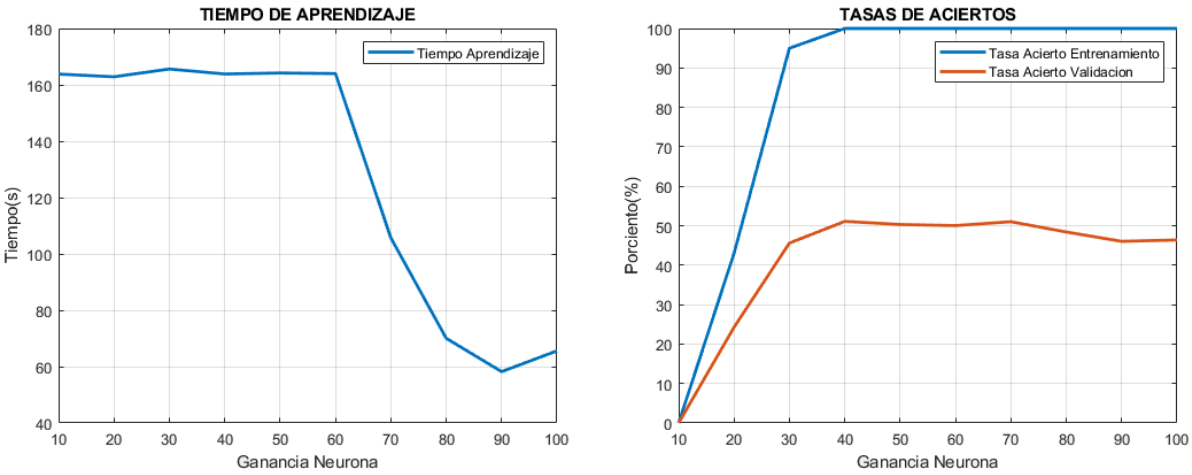


Figura 21. Comportamiento de la RNA con modelo SQS para diferentes valores de ganancia de la neurona.

Para el caso de la ganancia de la neurona ocurre un comportamiento opuesto al de la red con modelo SS. La red con este modelo de sinapsis necesita altas ganancias en las neuronas para su optimización y para que el algoritmo de aprendizaje converja en menor tiempo. El problema de saturación de las redes con este modelo podría resolverse con la disminución de esta ganancia.

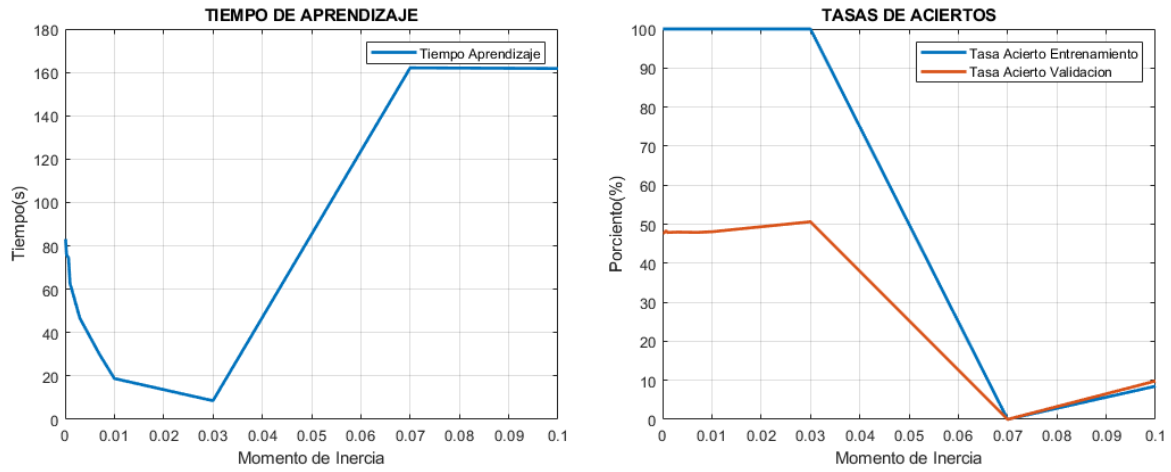


Figura 22. Comportamiento de la RNA con modelo SQS para diferentes valores de momento de inercia.

Corroborando el comportamiento obtenido de la tasa de aprendizaje, se manifiesta una convergencia más acelerada y mejor desempeño final de la red si se eligen valores pequeños de momento de inercia que resultaría en pasos pequeños en el proceso de aprendizaje.

4.2 RECONOCIMIENTO

Luego de optimizados los hiperparámetros del aprendizaje y entrenadas las redes con una misma topología, cada red neuronal fue sometida a una validación con el conjunto de datos de prueba, donde se recolectó el porcentaje de reconocimiento. Las redes neuronales con modelos SS, SASv2 y SQS lograron aprender y generalizar con un rendimiento semejante, mientras que el modelo SASv1 resultó no ser funcional (Tabla 5).

Tabla 5. Resultados de las redes neuronales optimizadas con diferentes modelos de sinapsis en el reconocimiento de patrones del conjunto de validación.

Modelo de Sinapsis		Resultados		
		Tiempo de Aprendizaje (s)	Tasa de Aciertos CE (%)	Tasa de Aciertos CV (%)
$SS_{ij}^{L} = O_i^{L-1} * W_{ij}^{L}$	----	13.89	100	58.27
$SASv1_{ij}^{L} = 1 - \left\|\left\|W_{ij}^{L}\right\| - O_i^{L-1}\right\| / K$	K=1.0	---	---	---
$SASv2_{ij}^{L} = sign(W_{ij}^{L}) * O_i^{L-1} * (1 - \frac{\left\|\left\|W_{ij}^{L}\right\| - O_i^{L-1}\right\|}{K})$	K=1.0	32.75	100	58.78
$SQS_{ij}^{L} = 0.1666 - \left(O_i^{L-1} - W_{ij}^{L}\right)^2 / K$	K=1.0	45.21	100	60.77

Hasta aquí, sólo se han optimizado y analizado los hiperparámetros que intervienen en el aprendizaje, pero no se ha estudiado la influencia de la selectividad añadida a los nuevos modelos de sinapsis en el reconocimiento final de la red. Esta selectividad en los modelos no sólo podría controlar grado de importancia de la comparación realizada en los modelos, sino también, presentar mejoras en el funcionamiento ante la variación de factores no controlados como el ruido en los patrones.

Las señales, en la realidad, se encuentran bajo la influencia de otras señales no deseadas e incluso en la acción de procesarlas suelen introducirse perturbaciones no favorables. Estas perturbaciones contaminan la señal de entrada como un ruido no deseado, que no guarda relación alguna con la señal útil. Las fuentes de ruido en los experimentos pueden llegar a alterar la salida de las redes, por lo que es necesario añadirlas al estudio. Se decidió añadir ruido blanco con distribución uniforme, obtenido a partir de la función "*rand*" de Matlab y reducido su valor máximo a 0.3, a las imágenes para realizar el reconocimiento nuevamente y medir el comportamiento de las redes con diferentes modelos ante este escenario; teniendo en cuenta además el factor selectividad.

4.2.1 RECONOCIMIENTO CON MODELO DE SINAPSIS ESTANDAR

Luego de añadido el ruido a los patrones de entrada y sometida al entrenamiento nuevamente con estas alteraciones, la variante de red con modelo estándar consiguió un correcto reconocimiento del 46.61 % de los patrones, lo que representa una notable disminución de su anterior tasa de aciertos.

Como las imágenes de dígitos que utilizamos anteriormente presentaban un fondo de valor numérico "0" el producto escalar que se realiza en el modelo SS resulta favorecido. Al añadir ruido, cada uno de estos valores de fondo dejan de ser cero, y las multiplicaciones con sus pesos producen gran cantidad de excitaciones que contribuyen a la suma acumulada de la neurona, aumentan la probabilidad de saturación de la misma y disminuyen el rendimiento de la red como fue comprobado.

Para este modelo no se analizó la influencia de la selectividad pues no contiene esta característica.

4.2.2 RECONOCIMIENTO CON MODELO DE SINAPSIS ABSOLUTA SELECTIVA v2

Para ilustrar la atribución en el reconocimiento del factor K, presente en el modelo SASv2, es necesario ejecutar la red neuronal de este modelo para un intervalo de valores del mismo; el intervalo seleccionado en este trabajo fue de [0.2, 4] con paso 0.2.

Ante la presencia de ruido, tanto en el entrenamiento como en el reconocimiento, esta red se mostró más sensible con una disminución al 18.50% de acierto con el factor K=1.0, valor con el cual la red fue evaluada en la simulación anterior. Este modelo presenta la misma desventaja del modelo SS con la eliminación del fondo nulo, que resultaba beneficioso para el rendimiento de las redes neuronales con estos modelos.

Sin embargo, en el reconocimiento para el valor de K=2.0 la red neuronal no solo aumentó su selectividad, sino también su tasa de acierto, alcanzando un 41.95% de efectividad. Esta mejora atribuida por el factor K tiene su explicación en que la inserción de gran cantidad de valores ruidosos requiere una menor selectividad de la neurona, ya que aumentan las diferencias y un espectro más amplio evita la eliminación de valores cuyas contribuciones pueden ser importantes para la clasificación.

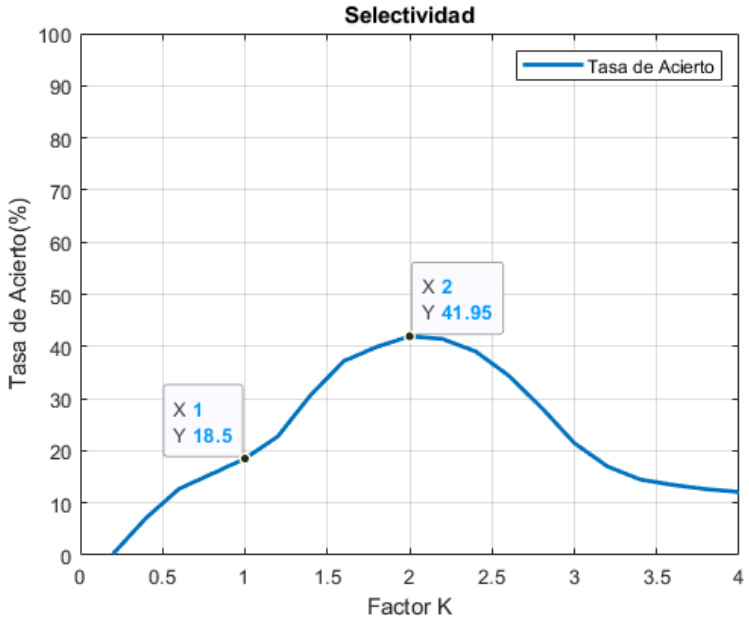


Figura 23. Comportamiento de la red neuronal con modelo SASv2 ante diferentes valores del factor K y presencia de ruido en las imágenes.

4.2.3 RECONOCIMIENTO CON MODELO DE SINAPSIS CUADRATICA SELECTIVA

Por otro lado, el desempeño de la red neuronal con modelo SSQ, aunque también se ve afectado por este ruido aditivo, devuelve resultados mucho más positivos que las anteriores redes y con una menos notable diferencia en la comparación con su anterior simulación ante patrones sin ruido. Este modelo manifiesta mayor robustez ante perturbaciones y una gran sensibilidad ante variaciones del factor K, con una efectividad máxima de 50.27% para el valor de K=1.0 con el cual fue entrenada.

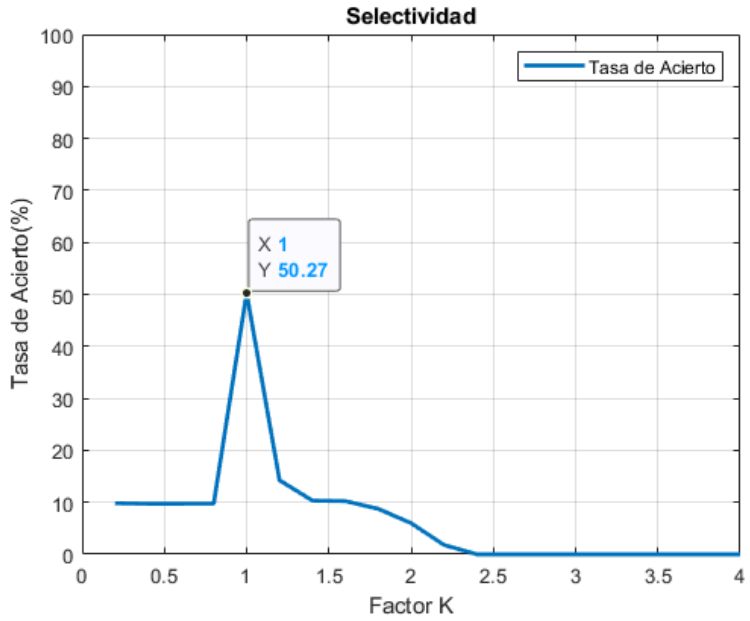


Figura 24. Comportamiento de la red neuronal con modelo SQS ante diferentes valores del factor K y presencia de ruido en las imágenes.

Después de observar que el mejor desempeño de esta red se consigue con el valor del factor K con el cual fue entrenada, se procedió a simular el entrenamiento y reconocimiento de esta red neuronal con otras dos variantes, K=0.9 y K=2.0. Para el valor de K=2.0 no se muestra mejora alguna, pues para varias simulaciones se obtienen resultados ligeramente menores que con K=1.0. Por otro lado, el modelo SQS obtiene un mejor funcionamiento cuando aumentamos la selectividad de la neurona con K=0.9, donde la neurona discrimina los pequeños valores de entrada y por tanto gran cantidad de ruido. Destacar que la disminución excesiva del factor K conlleva a la eliminación de valores que pueden ser importantes para la clasificación, así como la fuerte influencia de la aleatoriedad del ruido en la tasa final de aciertos.

4.3 COMPARACIÓN

Puede resultar abrumador seleccionar el mejor modelo de sinapsis para implementar en este problema de reconocimiento con una red neuronal, teniendo en cuenta la infinidad de posibilidades que son necesarias ejecutar para determinarlo y los competitivos resultados alcanzados con algunos de los modelos en determinados escenarios. En el afán de facilitar la comparación se agruparon todos los resultados obtenidos anteriormente en las diferentes etapas del proceso de reconocimiento de cada red neuronal, conformándose así la Tabla 6.

Tabla 6. Resumen de los resultados obtenidos en las simulaciones para cada modelo de red neuronal.

Modelo de Sinapsis	Aprendizaje					Reconocimiento		Resultados
	Factor K	Ruido	Hiperparámetros			Factor K	Ruido	Tasa de Aciertos CV (%)
			tasa aprendizaje	ganancia neurona	momento de inercia			
SS	----	No	0,07	40	0,3	----	No	58.27
SASv1	K=1.0	No	----	----	----	----	No	---
SASv2	K=1.0	No	0,7	40	0,4	K=1.0	No	58.78
SASv2	K=2.0	No	0,7	40	0,4	K=2.0	No	58.89
SQS	K=0.9	No	0,0007	80	0,03	K=0.9	No	60.53
SQS	K=1.0	No	0,0007	80	0,03	K=1.0	No	60.77
SQS	K=2.0	No	0,0007	80	0,03	K=2.0	No	59.98
SS	----	Si	0,07	30	0,3	----	Si	46,61
SASv2	K=1.0	Si	0,7	20	0,4	K=1.0	Si	18,5
SASv2	K=1.0	Si	0,7	20	0,4	K=2.0	Si	41,95
SASv2	K=2.0	Si	0,7	20	0,4	K=2.0	Si	43,85
SQS	K=0.9	Si	0,0007	80	0,03	K=0.9	Si	52,67
SQS	K=1.0	Si	0,0007	80	0,03	K=1.0	Si	51,63
SQS	K=2.0	Si	0,0007	80	0,03	K=2.0	Si	51,32

Si se observan los resultados, el lógico seleccionar al modelo SQS como candidato a modelo con mejor desempeño ante estos patrones de reconocimiento en los dos escenarios, tanto con ruido como sin él. Aunque pequeñas disminuciones del factor K de este modelo no siempre

conllevan a una mejoría absoluta, es cierto que, ante ruido, se puede incrementar la tasa de aciertos.

A medida que aumenta la magnitud del ruido le es más difícil a las redes neuronales establecer las relaciones entre las entradas y las salidas, muchas características de las imágenes son alteradas y la eficiencia de las redes disminuye como se muestra en la siguiente figura.

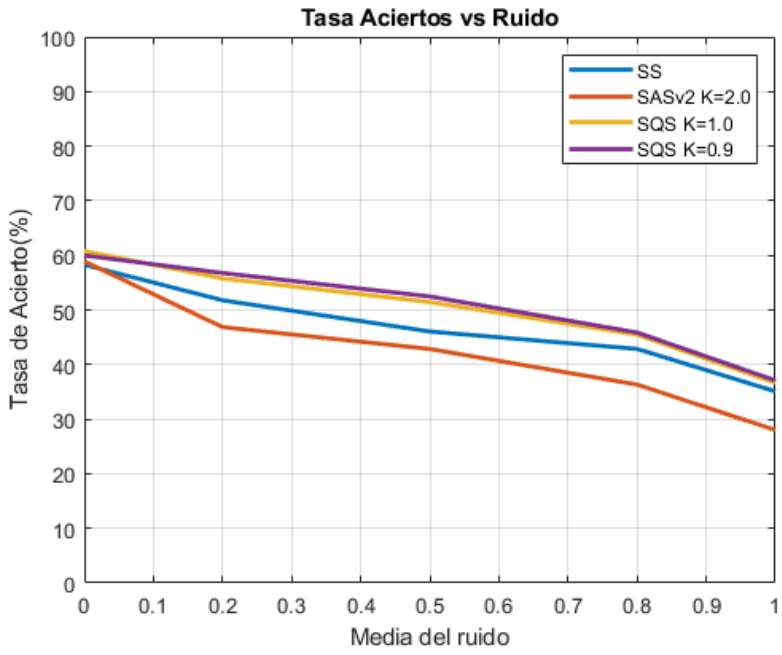


Figura 25. Comportamiento de diferentes diseños de redes neuronales ante la variación del ruido en los patrones de la base de datos.

La red neuronal con modelo SQS y factor K=0.9 muestra la más amplia capacidad de generalización en cuanto a que puede reconocer un patrón a pesar de estar contaminado con ruido y contener diferencias con respecto al patrón original.

CONCLUSIONES

Por los resultados expuestos hasta aquí, se puede concluir que la implementación de los nuevos diseños de sinapsis SASv2 y SQS, además de resolver el problema de saturación de la sinapsis estándar ante la presencia de gran cantidad de valores de entrada altos, poseen las ventajas de ser modelos cuyas salidas están en correspondencia con la similitud entre las señales de entrada y sus pesos, lo que puede ser adecuado y conveniente en problemas de clasificación.

La definición de nuevos diseños de sinapsis llevó a la conformación de nuevas redes neuronales, cuyo algoritmo de aprendizaje backpropagation fue primeramente generalizado y luego particularizado para cada una de las redes neuronales.

Utilizando la base de datos MNIST de dígitos escritos a mano fueron sometidas a entrenamiento las redes neuronales y analizados algunos de sus hiperparámetros para su optimización, devolviendo resultados finales similares con los modelos SS, SASv2 y SQS, mientras que, con el modelo SASv1 la red neuronal no consiguió realizar el ajuste de sus pesos durante el entrenamiento, por lo que resultó ser no funcional en el proceso de reconocimiento.

Esta problemática de reconocimiento fue obstaculizada con la incorporación de ruido blanco en las imágenes, alcanzando mayor porcentaje de acierto nuevamente la red neuronal con el modelo SQS y una mayor robustez ante el incremento del mismo.

Como fruto de esta labor investigadora, se ha presentado el artículo “Selective Quadratic Synapses (SQS)” [18] en la revista científica IEEE Transactions on Neural Networks and Learning Systems encargada de publicar artículos técnicos relacionados con la teoría, el diseño y las aplicaciones de redes neuronales y sistemas de aprendizaje relacionados.

Esta investigación presenta resultados prometedores, por lo que se continuará con el perfeccionamiento de estos nuevos modelos y su utilización en otras problemáticas.

REFERENCIAS

[1]. Martín-del-Brío, Bonifacio & Sanz, Alfredo, "Redes neuronales y sistemas borrosos" / B. Martín del Brío, A. Sanz Molina; pról. de Lotfi A. Zadeh, 2006.

[2]. [url=https://www.frontiersin.org/research-topics/4817/artificial-neural-networks-as-models-of-neural-information-processing «Artificial Neural Networks as Models of Neural Information Processing | Frontiers Research Topic»].

[3]. M. Paz Sesmero Lorente, "Diseño análisis y evaluación de conjuntos de clasificadores basados en redes de neuronas", Septiembre-2012: http://e-archivo.uc3m.es/bitstream/10016/16177/1/tesis_paz_sesmero_lorente_2012.pdf

[4]. J. Botia, H. Sarmiento y C. Isaza., "Redes Neuronales Artificiales de Base Radial como Clasificador Difuso: Una Aplicación en Diagnóstico Médico", 2013: http://www.academia.edu/1364869/Redes_Neuronales_Artificiales_de_Base_Radial_como_Clasificador_Difuso_Una_Aplicacion_al_Diagnostico_Medico_

[5]. HAYKIN, S., "Neural Networks – A Comprehensive Foundation". Prentice-Hall, Delhi, 2001.

[6]. S.K. Pal and P. K. Srrimani, “Neurocomputing: motivation, models and hybridization”, Computer Magazine, IEEE Computer Society, Vol. 29, No. 3, March 1996.

[7]. Rakitianskaia y A. Engelbrecht, "Measuring Saturation in Neural Networks", 2015 IEEE Symposium Series on Computational Intelligence, Ciudad del Cabo, 2015, pp. 1423-1430, doi: 10.1109/SSCI.2015.202.

[8]. X. Glorot e Y. Bengio, "Understanding the difficulti of training deep feedforward neural networks", International conference on artificial intelligence and statistics, pp. 249-256, 2010.

[9]. A. Rakitianskaia y A. P. Engelbrecht, "Training high-dimensional neural networks with cooperative particle swarm optimiser", International Joint Conference on Neural Networks (IJCNN), pp. 4011-4018, 2014.

[10]. J. M. Quero y P. J. Quero, "Stochastic Neural Interface with Selective Synapse", *Conferencia Conjunta Internacional sobre Redes Neuronales (IJCNN) de 2018*, pp. 1-6, doi: 10.1109/IJCNN.2018.8489243.

[11]. Schmidhuber, J. (2015). "Aprendizaje profundo en redes neuronales: una visión general". Redes neuronales. 61: 85–117

[12]. Goodfellow, Ian; Bengio, Yoshua; Courville, Aaron, 2016. "6.5 Back-Propagation y otros algoritmos de diferenciación". Aprendizaje Profundo. MIT Press. ISBN 9780262035613.

[13]. Rumelhart; Hinton; Williams, 1986. "Representaciones de aprendizaje mediante errores de propagación". 323 (6088): 533–536.

[14]. Dreyfus, Stuart, 1962. "La solución numérica de los problemas de variación". Revista de Análisis Matemático y Aplicaciones. 5 (1): 30–45.

[15]. [url=http://yann.lecun.com/exdb/mnist/]

[16]. S. Lawrence, A.C. Tsoi y A. D. Atrás, "Function approximation with neural networks and local methods: bias variance and smoothness", Proceedings of the Australian Conference on Neural Networks, pp.16-21, 1996.

[17]. K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink y J. Schmidhuber, "LSTM: A Search Space Odyssey", en IEEE Transactions on Neural Networks and Learning Systems, vol. 28, No. 10, pp. 2222-2232, octubre de 2017, doi: 10.1109/TNNLS.2016.2582924.

[18]. José M. Quero y Gerardo Gómez Pérez, “Selective Quadratic Synapses (SQS)”, en IEEE Transactions on Neural Networks and Learning Systems. En revision.

ANEXOS

Anexo 1. Simulación de salida de neuronas con diferentes modelos de sinapsis ante ráfaga de valores pequeños e iguales de W y O.

Ind.	W	O	SS	SASv1	SASv2	SQS	
1	0,1	0,1	0,01	1	0,1	0,1666	
2	0,1	0,1	0,01	1	0,1	0,1666	
3	0,1	0,1	0,01	1	0,1	0,1666	
4	0,1	0,1	0,01	1	0,1	0,1666	
5	0,1	0,1	0,01	1	0,1	0,1666	
6	0,1	0,1	0,01	1	0,1	0,1666	
7	0,1	0,1	0,01	1	0,1	0,1666	
8	0,1	0,1	0,01	1	0,1	0,1666	
9	0,1	0,1	0,01	1	0,1	0,1666	
10	0,1	0,1	0,01	1	0,1	0,1666	
11	0,1	0,1	0,01	1	0,1	0,1666	
12	0,1	0,1	0,01	1	0,1	0,1666	
…							
96	0,1	0,1	0,01	1	0,1	0,1666	
97	0,1	0,1	0,01	1	0,1	0,1666	
98	0,1	0,1	0,01	1	0,1	0,1666	
99	0,1	0,1	0,01	1	0,1	0,1666	
100	0,1	0,1	0,01	1	0,1	0,1666	
101	0,1	0,1	0,01	1	0,1	0,1666	
102	0,1	0,1	0,01	1	0,1	0,1666	
103	0,1	0,1	0,01	1	0,1	0,1666	
104	0,1	0,1	0,01	1	0,1	0,1666	
105	0,1	0,1	0,01	1	0,1	0,1666	
106	0,1	0,1	0,01	1	0,1	0,1666	
107	0,1	0,1	0,01	1	0,1	0,1666	
108	0,1	0,1	0,01	1	0,1	0,1666	
109	0,1	0,1	0,01	1	0,1	0,1666	
110	0,1	0,1	0,01	1	0,1	0,1666	
111	0,1	0,1	0,01	1	0,1	0,1666	
112	0,1	0,1	0,01	1	0,1	0,1666	
113	0,1	0,1	0,01	1	0,1	0,1666	
			1,13	113	11,3	18,8258	**neti**
	0,1	0,1	0,01	1	0,1	0,1666	**media**
			0,706822	1	0,999984	1	**salida neurona**

Anexo 2. Simulación de salida de neuronas con diferentes modelos de sinapsis ante ráfaga de valores grandes e iguales de W y O.

Ind.	W	O	SS	SASv1	SASv2	SQS	
1	0,9	0,9	0,81	1	0,9	0,1666	
2	0,9	0,9	0,81	1	0,9	0,1666	
3	0,9	0,9	0,81	1	0,9	0,1666	
4	0,9	0,9	0,81	1	0,9	0,1666	
5	0,9	0,9	0,81	1	0,9	0,1666	
6	0,9	0,9	0,81	1	0,9	0,1666	
7	0,9	0,9	0,81	1	0,9	0,1666	
8	0,9	0,9	0,81	1	0,9	0,1666	
9	0,9	0,9	0,81	1	0,9	0,1666	
10	0,9	0,9	0,81	1	0,9	0,1666	
11	0,9	0,9	0,81	1	0,9	0,1666	
12	0,9	0,9	0,81	1	0,9	0,1666	
…							
96	0,9	0,9	0,81	1	0,9	0,1666	
97	0,9	0,9	0,81	1	0,9	0,1666	
98	0,9	0,9	0,81	1	0,9	0,1666	
99	0,9	0,9	0,81	1	0,9	0,1666	
100	0,9	0,9	0,81	1	0,9	0,1666	
101	0,9	0,9	0,81	1	0,9	0,1666	
102	0,9	0,9	0,81	1	0,9	0,1666	
103	0,9	0,9	0,81	1	0,9	0,1666	
104	0,9	0,9	0,81	1	0,9	0,1666	
105	0,9	0,9	0,81	1	0,9	0,1666	
106	0,9	0,9	0,81	1	0,9	0,1666	
107	0,9	0,9	0,81	1	0,9	0,1666	
108	0,9	0,9	0,81	1	0,9	0,1666	
109	0,9	0,9	0,81	1	0,9	0,1666	
110	0,9	0,9	0,81	1	0,9	0,1666	
111	0,9	0,9	0,81	1	0,9	0,1666	
112	0,9	0,9	0,81	1	0,9	0,1666	
113	0,9	0,9	0,81	1	0,9	0,1666	
			91,53	113	101,7	18,8258	**neti**
	0,9	0,9	0,81	1	0,9	0,1666	**media**
			1	1	1	1	**salida neurona**

Anexo 3. Simulación de salida de neuronas con diferentes modelos de sinapsis ante ráfaga de valores de W y O con gran diferencia entre ellos.

Ind.	W	O	SS	SASv1	SASv2	SQS	
1	0,9	0,1	0,09	0,2	0,02	-0,4734	
2	0,9	0,1	0,09	0,2	0,02	-0,4734	
3	0,9	0,1	0,09	0,2	0,02	-0,4734	
4	0,9	0,1	0,09	0,2	0,02	-0,4734	
5	0,9	0,1	0,09	0,2	0,02	-0,4734	
6	0,9	0,1	0,09	0,2	0,02	-0,4734	
7	0,9	0,1	0,09	0,2	0,02	-0,4734	
8	0,9	0,1	0,09	0,2	0,02	-0,4734	
9	0,9	0,1	0,09	0,2	0,02	-0,4734	
10	0,9	0,1	0,09	0,2	0,02	-0,4734	
11	0,9	0,1	0,09	0,2	0,02	-0,4734	
12	0,9	0,1	0,09	0,2	0,02	-0,4734	
…							
96	0,9	0,1	0,09	0,2	0,02	-0,4734	
97	0,9	0,1	0,09	0,2	0,02	-0,4734	
98	0,9	0,1	0,09	0,2	0,02	-0,4734	
99	0,9	0,1	0,09	0,2	0,02	-0,4734	
100	0,9	0,1	0,09	0,2	0,02	-0,4734	
101	0,9	0,1	0,09	0,2	0,02	-0,4734	
102	0,9	0,1	0,09	0,2	0,02	-0,4734	
103	0,9	0,1	0,09	0,2	0,02	-0,4734	
104	0,9	0,1	0,09	0,2	0,02	-0,4734	
105	0,9	0,1	0,09	0,2	0,02	-0,4734	
106	0,9	0,1	0,09	0,2	0,02	-0,4734	
107	0,9	0,1	0,09	0,2	0,02	-0,4734	
108	0,9	0,1	0,09	0,2	0,02	-0,4734	
109	0,9	0,1	0,09	0,2	0,02	-0,4734	
110	0,9	0,1	0,09	0,2	0,02	-0,4734	
111	0,9	0,1	0,09	0,2	0,02	-0,4734	
112	0,9	0,1	0,09	0,2	0,02	-0,4734	
113	0,9	0,1	0,09	0,2	0,02	-0,4734	
			10,17	22,6	2,26	-53,4942	**neti**
	0,9	0,1	0,09	0,2	0,02	-0,4734	**media**
			0,999951	1	0,881843	4,56E-24	**salida neurona**

Anexo 4. Simulación de valores aleatorios con distribución uniforme de W y O para la comprobación del cálculo del valor medio estadístico de esta diferencia cuadrática.

Ind.	W	O	(W - O) ^2	
1	0,34260840	0,44005976	0,00949677	
2	0,74214341	0,20804564	0,28526043	
3	0,43899739	0,62314912	0,03391186	
4	0,67140745	0,00364652	0,44590466	
5	0,66255958	0,51422409	0,02200342	
6	0,48137378	0,82064020	0,11510170	
7	0,00918852	0,96091272	0,90577896	
8	0,74195308	0,52411818	0,04745204	
9	0,93294196	0,60641663	0,10661879	
10	0,77812967	0,46374627	0,09883692	
11	0,04829646	0,61502544	0,32118174	
12	0,69614908	0,32515582	0,13763601	
…				
96	0,53539275	0,67072546	0,01831494	
97	0,37661099	0,84993159	0,22403239	
98	0,90817552	0,34865085	0,31306786	
99	0,31205238	0,51196364	0,03996451	
100	0,96378055	0,69378501	0,07289759	
101	0,92593246	0,13793124	0,62094591	
102	0,92846009	0,44880293	0,23007099	
103	0,56415605	0,74096359	0,03126091	
104	0,29580622	0,24518412	0,00256260	
105	0,14462065	0,51981794	0,14077301	
106	0,27116839	0,38886704	0,01385297	
107	0,70624515	0,52239876	0,03379950	
108	0,62415630	0,95607639	0,11017095	
109	0,83119549	0,53315388	0,08882880	
110	0,62680310	0,46901393	0,02489742	
111	0,98391411	0,50088576	0,23331639	
112	0,02394324	0,63404420	0,37222319	
113	0,90761338	0,46026696	0,20011882	
			0,05875800	**neti**
	0,53447575	0,47605113	**0,16668002**	**media**
			0,47321517	**salida neurona**