Universidad del Bío-Bío, Chile

Facultad de Ciencias Empresariales

Departamento de Sistemas de Información

Una contribución al estudio de la clasificación de imágenes empleando aprendizaje profundo sobre conjuntos pequeños de datos: desafíos, técnicas, implementación y evaluación

Tesis presentada por Gonzalo Miranda Cabrera

para obtener el grado de Magíster en Ciencias de la Computación

Dirigida por Clemente Rubio Manzano

2021

Agradecimientos

A mi madre.

Resumen

La Inteligencia Artificial está revolucionando nuestra vida diaria, está presente en multitud de aplicaciones que nos ayudan en la toma de decisiones. En los últimos años, debido al aumento de la capacidad computacional de los sistemas informáticos y a la generación de grandes volúmenes de datos, se ha producido un gran avance en la investigación y el desarrollo del aprendizaje automático dando lugar a grandes progresos en el Aprendizaje Profundo y a su consolidación como una de las técnicas Inteligencia Artificial más empleadas en la actualidad.

Uno de los requisitos fundamentales para que los datos puedan utilizarse para generar buenos modelos de aprendizaje automático es que deben etiquetarse correctamente. Sin embargo, el proceso de etiquetado de una gran cantidad de datos puede resultar costoso y lento dependiendo de qué tan grande sea la cantidad de datos a etiquetar. Otro problema que surge de la necesidad de contar con grandes cantidades de datos, es que en algunos campos, como la medicina, los datos son escasos ya que están ligados a la cantidad de pacientes tratados, lo cual limita la cantidad de datos disponibles; o en campos como la industria y la ciencia de materiales existe una dificultad de recolección debido a los procesos relacionados con su obtención.

Ante esta problemática en la literatura se plantean diferentes técnicas que intentan mitigar la deficiencia en cantidad de ejemplos para entrenamiento. El objetivo de esta tesis es estudiar el desempeño de las diferentes técnicas propuestas para trabajar con pequeñas cantidades de datos empleando Aprendizaje Profundo. Además, se agregan a la comparación algoritmos de aprendizaje automático clásico, para contrastar los resultados con ese tipo de algoritmos.

Para ello, se realiza una búsqueda en la literatura para encontrar las técnicas de aprendizaje profundo utilizadas para clasificación de imágenes con limitados ejemplos. Luego, se han imple-

mentado tres algoritmos clásicos, Árbol de Decisión, Bosque Aleatorio y Máquina de Soporte de Vectores; y nueve algoritmos de Redes Neuronales Profundas. Entre algoritmos clásicos, técnicas de aprendizaje profundo y combinaciones se completa un total de 13 algoritmos diferentes. Estos fueron entrenados sobre cuatro subconjuntos de diferentes tamaños (10, 50, 250 y 500 imágenes por clase) de los conjuntos de datos MNIST, Fashion MNIST y CIFAR-10, los cuales fueron escogidos de manera aleatoria desde el conjunto de entrenamiento de cada conjunto de datos. En total se realizaron 156 experimentos, ya que cada algoritmo se entrena en cada uno de los subconjuntos para cada uno de los tres conjuntos utilizados.

Finalmente, los resultados muestran que si existe una técnica o combinación de técnicas que obtiene mejor desempeño con conjuntos pequeños de datos, pero esta técnica o combinación es diferente para cada conjunto de datos. Sin embargo, Dropout, Agrupación de promedio global, Transferencia de aprendizaje y el Ensamblaje de múltiples modelos son técnicas con las que se recomienda experimentar ya que son las técnicas que muestran los mayores incrementos en los resultados.

Palabras Clave — Inteligencia Artificial, Aprendizaje Automático, Aprendizaje Profundo, Conjunto pequeño de datos, Clasificación de imágenes

Abstract

In recent years, the increase in computer systems'computational capacity and the generation of a large volume of data have allowed significant advances in machine learning and deep learning consolidation as some of the most widely used Artificial Intelligence techniques today.

Good machine and deep learning models require a correctly labelled set of data. However, two critical problems can appear in this process: i) the task of labelling a large amount of data is expensive and/or time-consuming; ii) some fields, like the medical field, lack big amounts of data because is direcly linked with the amount of patiens treated, so this fields show data scarcity. On the other hand, there are some fields, like industry and materials science, where the processes associated with data collection may be hard or impossible to carry out depending on the nature of said process.

To face this problem, diferent techniques are proposed in the literature that try to dimish the effect of this problem on the overall algorithm's performance. This master's thesis aims to study the different available techniques and perform a complete experimentation to analyse which technique or combination of them improves the performance of Deep Learning algorithms. Simultaneously, a comparison with classical machine learning techniques will be performed to contrast the results obtained using both kinds of techniques when working with small datasets.

With this objective in mind, a literature search is performed to find Deep Learning techniques that were used for image classificaction on small datasets. Then, three classical algorithms are implemented, Decision Tree, Random Forest and Vector Support Machine; and nine Deep Neural Network algorithms. Between classic algorithms, Deep Neural Networks and technique combinations a total of 13 different algorithms is completed. All Algorithms were trained on four

subsets of diferent sizes (10, 50, 250 and 500 images per class) of the MNIST, Fashion MNIST and CIFAR-10 datasets. Each image for the subsets were chosen randomly from the training set of each dataset. In total, 156 experiments are carried out, since each algorithm is trained in each of the subsets for each of the three sets used.

Finally, the results show that there is a technique or combination of techniques that performs better with small datasets, but this technique or combination of them is different for each dataset. However, Dropout, Global Average Pooling, Transfer Learning, and Multiple Model Ensembles are techniques that we recommend experimenting with as they are the techniques that show the largest increases in results.

Keywords — Artificial Intelligence, Deep Learning, Small Dataset, Machine Learning, Image Classification

Índice general

1 Introducción **1**

1.1 Problema . . . 3

1.1.1 Soluciones propuestas en la literatura . . . 4

1.2 Hipótesis y objetivos . . . 4

1.2.1 Hipótesis . . . 4

1.2.2 Objetivos . . . 5

1.3 Organización de la tesis . . . 6

2 Aprendizaje automático **7**

2.1 Entrenamiento . . . 8

2.2 Métricas para clasificación . . . 10

2.3 Funciones de costo . . . 13

2.3.1 Entropía cruzada Categórica. . . . 13

3 Algoritmos de aprendizaje automático **16**

3.1 Árbol de decisión . . . 16

3.1.1 Bosque aleatorio . . . 17

3.2 Máquina de soporte de vectores . . . 18

3.3 Redes neuronales artificiales . . . 20

3.3.1 Perceptrón multicapa . . . 22

3.4 Redes neuronales profundas . . . 22

3.4.1 Capa densa 24
3.4.2 Capa convolucional 24
3.4.3 Capa aplanar 26
3.4.4 Capa de ampliación 27
3.4.5 Capa de agrupación 27
3.5 Aprendizaje profundo con conjuntos pequeños de datos 28

4 Técnicas de aprendizaje profundo para trabajar con conjuntos pequeños de datos 30
4.1 Regularización 30
4.1.1 Capa dropout 30
4.1.2 Normalización por lote 31
4.2 Capa de agrupación promedio global 32
4.3 Convolución dilatada 33
4.4 Función de costo similitud de coseno 34
4.5 Tasa de aprendizaje cíclica 35
4.6 Aumento de la cantidad datos 37
4.7 Transferencia de aprendizaje 39
4.8 Ensamblaje de múltiples modelos 40

5 Clasificación de imágenes con conjuntos pequeños de datos 42

6 Experimentación y resultados 52
6.1 Experimentación 52
6.1.1 Conjuntos de datos 52
6.1.2 Algoritmos y técnicas 55
6.1.3 Métricas a utilizar para la evaluación 58
6.1.4 Configuración del sistema empleado 59
6.2 Resultados 59
6.2.1 Subconjuntos de MNIST 59

6.2.2 Subconjuntos de Fashion MNIST 65
6.2.3 Subconjuntos de CIFAR-10 70
6.2.4 Discusión de los resultados 75

Aportes, conclusiones y trabajo futuro **77**
Aportes 77
Conclusiones 78
Trabajo futuro 79

Referencias **80**

A Configuración de los algoritmos **88**
A.1 Configuración de algoritmos entrenados sobre subconjuntos de MNIST 88
A.1.1 Árbol de decisión 88
A.1.2 Bosque Aleatorio 89
A.1.3 Máquina de Soporte de Vectores 90
A.1.4 Red Neuronal Profunda 91
A.1.5 Red Neuronal Profunda con Dropout 92
A.1.6 Red Neuronal Profunda con capa Agrupación Promedio Global 94
A.1.7 Red Neuronal Profunda con Normalización por lotes 96
A.1.8 Red Neuronal Profunda con función de costo Similitud de Coseno 98
A.1.9 Red Neuronal Profunda con Convolución Dilatada 100
A.1.10 Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje . 102
A.1.11 Red Neuronal Profunda con combinación de técnicas (C1) 104
A.1.12 Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos (C2) 106
A.1.13 Ensamblaje de todas las Redes Neuronales Profundas 108
A.2 Configuración de algoritmos entrenados sobre subconjuntos de Fashion MNIST 109
A.2.1 Árbol de decisión 109
A.2.2 Bosque Aleatorio 110

A.2.3 Máquina de Soporte de Vectores 111
A.2.4 Red Neuronal Profunda 111
A.2.5 Red Neuronal Profunda con Dropout 113
A.2.6 Red Neuronal Profunda con capa Agrupación Promedio Global 115
A.2.7 Red Neuronal Profunda con Normalización por lotes 117
A.2.8 Red Neuronal Profunda con función de costo Similitud de Coseno 119
A.2.9 Red Neuronal Profunda con Convolución Dilatada 121
A.2.10 Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje . 123
A.2.11 Red Neuronal Profunda con combinación de técnicas (C1) 125
A.2.12 Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos (C2) 128
A.2.13 Ensamblaje de todas las Redes Neuronales Profundas 130
A.3 Configuración de algoritmos entrenados sobre subconjuntos de CIFAR-10 . . . 131
A.3.1 Árbol de decisión 131
A.3.2 Bosque Aleatorio 131
A.3.3 Máquina de Soporte de Vectores 132
A.3.4 Red Neuronal Profunda 133
A.3.5 Red Neuronal Profunda con Dropout 135
A.3.6 Red Neuronal Profunda con capa Agrupación Promedio Global 137
A.3.7 Red Neuronal Profunda con Normalización por lotes 139
A.3.8 Red Neuronal Profunda con función de costo Similitud de Coseno 141
A.3.9 Red Neuronal Profunda con Convolución Dilatada 143
A.3.10 Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje . 145
A.3.11 Red Neuronal Profunda con combinación de técnicas (C1) 147
A.3.12 Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos (C2) 149
A.3.13 Ensamblaje de todas las Redes Neuronales Profundas 151

Lista de Figuras

1.1 Diagrama conceptual de Aprendizaje Profundo (Goodfellow et al., 2016). 2
1.2 Imágenes de animales correctamente etiquetadas. 3

2.1 Ejemplo de desempeño bajo ajuste, correctamente ajustado y sobreajuste. . . . 10
2.2 Matriz de confusión. 11
2.3 Ejemplo de Codificación One-hot para tres clases. 14
2.4 Diagrama de la composición de la función de costo Entropía cruzada Categórica. 15

3.1 Ejemplo de Árbol de decisión. 17
3.2 Diagrama de predicción por Bosque Aleatorio. 18
3.3 Separación de datos por SVM. 19
3.4 Separación de datos con gran margen. 19
3.5 Separación de datos altamente interrelacionados utilizando kernel trick. 20
3.6 Modelo de perceptrón propuesto por Rosenblatt (1958). 20
3.7 Modelo del perceptrón multicapa o Red Neuronal Prealimentada (Tang y Kwan, 1993). 22
3.8 Ejemplo de Red Neuronal Profunda con dos capas ocultas. 23
3.9 Ejemplo de operación de convolución. Imagen tomada de Dumoulin y Visin (2016). 25
3.10 Ejemplo de operación de convolución con padding. Imagen tomada de Dumoulin y Visin (2016). 26
3.11 Operación de convolución. Imagen tomada de Reynolds (2019). 26

3.12 Ejemplo de Capa de Agrupación Máxima con paso dos. Imagen tomada de Aurélien (2019). 28

4.1 Ejemplo de Dropout en una red neuronal. Imagen tomada de Srivastava et al. (2014). 31
4.2 Ejemplo de operación GAP. Imagen tomada de Peltarion (2019). 33
4.3 Ejemplos de campo receptivo con distintas tasas de dilatación. 34
4.4 Ejemplos de planificación de tasa de aprendizaje. 36
4.5 Ejemplo de CLR con los siguientes parámetros: lr_{min} de 0.0001, lr_{max} de 0.001 y s igual a 10. 36
4.6 Ejemplo de decadencia sobre CLR con los mismos parámetros que la Figura 4.5, δ igual a 10 y t_{max} igual a 80. 37
4.7 Ejemplo de aumento de datos (Data Augmentation). 38
4.8 Diagrama de transferencia de aprendizaje de la tarea X a la tarea Y. 39
4.9 Diagrama de Ensamblaje de Múltiples Modelos. 41

6.1 Ejemplos de los conjuntos de datos. 53
6.2 Diagrama de la experimentación realizada. 58
6.3 Gráfico de resultados de exactitud entre algoritmos evaluados sobre MNIST. . . 60
6.4 Gráfico de resultados de precisión entre algoritmos evaluados sobre MNIST. . . 61
6.5 Gráfico de resultados de recall entre algoritmos evaluados sobre MNIST. 62
6.6 Gráfico de resultados de F1 entre algoritmos evaluados sobre MNIST. 63
6.7 Gráfico de resultados de coeficiente de correlación de Matthews entre algoritmos evaluados sobre MNIST. 64
6.8 Gráfico de resultados de exactitud entre algoritmos evaluados sobre Fashion MNIST. 65
6.9 Gráfico de resultados de precisión entre algoritmos evaluados sobre Fashion MNIST. 66
6.10 Gráfico de resultados de recall entre algoritmos evaluados sobre Fashion MNIST. 67

6.11 Gráfico de resultados de F1 entre algoritmos evaluados sobre Fashion MNIST. . 68
6.12 Gráfico de resultados de coeficiente de correlación de Matthews entre algoritmos evaluados sobre Fashion MNIST. 69
6.13 Gráfico de resultados de exactitud entre algoritmos evaluados sobre CIFAR-10. 70
6.14 Gráfico de resultados de precisión entre algoritmos evaluados sobre CIFAR-10. 71
6.15 Gráfico de resultados de recall entre algoritmos evaluados sobre CIFAR-10. . . 72
6.16 Gráfico de resultados de F1 entre algoritmos evaluados sobre CIFAR-10. 73
6.17 Gráfico de resultados de coeficiente de correlación de Matthews entre algoritmos evaluados sobre CIFAR-10. 74

A.1 Gráficas de costo (Loss) y Exactitud (Accuracy) durante entrenamiento para Red Neuronal Profunda sobre MNIST. 92
A.2 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Dropout sobre MNIST. 94
A.3 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Capa Agrupación Promedio Global sobre MNIST. 96
A.4 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Normalización por Lotes sobre MNIST. 98
A.5 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con función de costo Similitud de Coseno sobre MNIST. 100
A.6 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Convolución Dilatada sobre MNIST. 102
A.7 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre MNIST. 104
A.8 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas sobre MNIST. 106
A.9 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre MNIST. 108

A.10 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda sobre Fashion MNIST. 113

A.11 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Dropout sobre Fashion MNIST. 115

A.12 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con capa Agrupación Promedio Global sobre Fashion MNIST. 117

A.13 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Normalización por Lotes sobre Fashion MNIST. 119

A.14 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con función de costo Similitud de Coseno sobre Fashion MNIST. 121

A.15 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Convolución Dilatada sobre Fashion MNIST. 123

A.16 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre Fashion MNIST. . . 125

A.17 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas sobre Fashion MNIST. 127

A.18 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre Fashion MNIST. 130

A.19 Gráficas de costo (Loss) y Exactitud (Accuracy) durante entrenamiento para Red Neuronal Profunda sobre CIFAR-10. 135

A.20 Gráficas de costo (Loss) y Exactitud (Accuracy) durante entrenamiento para Red Neuronal Profunda con Dropout sobre CIFAR-10. 137

A.21 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con capa Agrupación Promedio Global sobre CIFAR-10. 139

A.22 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Normalización por Lotes sobre CIFAR-10. 141

A.23 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con función de costo Similitud de Coseno sobre CIFAR-10. 143
A.24 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Convolución Dilatada sobre CIFAR-10. 145
A.25 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre CIFAR-10. 147
A.26 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas sobre CIFAR-10. 149
A.27 Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento de la cantidad de datos sobre CIFAR-10. 151

Índice de Tablas

4.1 Modelos disponibles en la biblioteca de código Keras (Chollet y otros, 2015). . . 40

5.1 Detalles de los conjuntos de datos utilizados en el trabajo de Barz y Denzler (2019). 47

5.2 Distribución de subconjuntos que utiliza D'souza et al. (2020). 49

5.3 Resumen de las técnicas que cada trabajo aplica para enfrentar el problema de contar con un conjunto pequeño de datos. 50

6.1 Comparación de resultados de exactitud sobre MNIST. 60

6.2 Comparación de resultados de precisión sobre MNIST. 61

6.3 Comparación de resultados de recall sobre MNIST. 62

6.4 Comparación de resultados de F1 sobre MNIST. 63

6.5 Comparación de resultados de coeficiente de correlación de Matthews sobre MNIST. 64

6.6 Comparación de resultados de exactitud sobre Fashion MNIST. 65

6.7 Comparación de resultados de precisión sobre Fashion MNIST. 66

6.8 Comparación de resultados de recall sobre Fashion MNIST. 67

6.9 Comparación de resultados de F1 sobre Fashion MNIST. 68

6.10 Comparación de resultados de coeficiente de correlación de Matthews sobre Fashion MNIST. 69

6.11 Comparación de resultados de exactitud sobre CIFAR-10. 70

6.12 Comparación de resultados de precisión sobre CIFAR-10. 71
6.13 Comparación de resultados de recall sobre CIFAR-10. 72
6.14 Comparación de resultados de F1 sobre CIFAR-10. 73
6.15 Comparación de resultados de coeficiente de correlación de Matthews sobre CIFAR-10. 74
6.16 Técnicas y algoritmos con resultados más altos en Exactitud, F1 y MCC para cada subconjunto de datos. 79

A.1 Tabla de parámetros para Árbol de decisión sobre MNIST. 89
A.2 Tabla de resultados de Árbol de decisión sobre MNIST. 89
A.3 Tabla de parámetros para Bosque Aleatorio sobre MNIST. 89
A.4 Tabla de resultados de Bosque Aleatorio sobre MNIST. 90
A.5 Tabla de parámetros para Máquina de Soporte de Vectores sobre MNIST. . . . 90
A.6 Tabla de resultados de Máquina de Soporte de Vectores sobre MNIST. 90
A.7 Tabla de parámetros para Red Neuronal Profunda sobre MNIST. 91
A.8 Tabla resumen de la Estructura de la Red Neuronal Profunda sobre MNIST. . 91
A.9 Resultados de la Red Neuronal Profunda sobre MNIST. 92
A.10 Tabla de parámetros para Red Neuronal Profunda con Dropout sobre MNIST. 93
A.11 Tabla resumen de la Estructura de Red Neuronal Profunda con Dropout sobre MNIST. 93
A.12 Resultados de la Red Neuronal Profunda con Dropout sobre MNIST. 94
A.13 Tabla de parámetros para Red Neuronal Profunda con Capa Agrupación Promedio Global sobre MNIST. 95
A.14 Tabla resumen de la Estructura de Red Neuronal Profunda con Capa Agrupación Promedio Global sobre MNIST. 95
A.15 Resultados de la Red Neuronal Profunda con Capa Agrupación Promedio Global sobre MNIST. 96
A.16 Tabla de parámetros para Red Neuronal Profunda con Normalización por Lotes sobre MNIST. 97

A.17 Tabla resumen de la Estructura de Red Neuronal Profunda con Normalización por Lotes sobre MNIST. 97

A.18 Resultados de la Red Neuronal Profunda con Normalización por Lotes sobre MNIST. 98

A.19 Tabla de parámetros para Red Neuronal Profunda con función de costo Similitud de Coseno sobre MNIST. 99

A.20 Tabla resumen de la Estructura de Red Neuronal Profunda con función de costo Similitud de Coseno sobre MNIST. 99

A.21 Resultados de la Red Neuronal Profunda con función de costo Similitud de Coseno sobre MNIST. 100

A.22 Tabla de parámetros para Red Neuronal Profunda con Convolución Dilatada sobre MNIST. 101

A.23 Tabla resumen de la Estructura de Red Neuronal Profunda con Convolución Dilatada sobre MNIST. 101

A.24 Resultados de la Red Neuronal Profunda con Convolución Dilatada sobre MNIST. 102

A.25 Tabla de parámetros para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre MNIST. 103

A.26 Tabla resumen de la Estructura de Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre MNIST. 103

A.27 Resultados de la Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre MNIST. 104

A.28 Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas sobre MNIST. 105

A.29 Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas sobre MNIST. 105

A.30 Resultados de la Red Neuronal Profunda con combinación de técnicas sobre MNIST. 106

A.31 Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre MNIST. . . 107
A.32 Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre MNIST. 107
A.33 Resultados de la Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre MNIST. 108
A.34 Resultados del ensamblaje de todas las Redes Neuronales Profundas sobre MNIST.109
A.35 Tabla de parámetros para Árbol de decisión sobre Fashion MNIST. 109
A.36 Tabla de resultados de Árbol de decisión sobre Fashion MNIST. 110
A.37 Tabla de parámetros para Bosque Aleatorio sobre Fashion MNIST. 110
A.38 Tabla de resultados de Bosque Aleatorio sobre Fashion MNIST. 110
A.39 Tabla de parámetros para la Máquina de Soporte de Vectores sobre Fashion MNIST. 111
A.40 Tabla de resultados de la Máquina de Soporte de Vectores sobre Fashion MNIST.111
A.41 Tabla de parámetros de la Red Neuronal Profunda sobre Fashion MNIST. . . . 112
A.42 Tabla resumen de la Estructura de Red Neuronal Profunda sobre Fashion MNIST.112
A.43 Resultados de la Red Neuronal Profunda sobre Fashion MNIST. 113
A.44 Tabla de parámetros para la Red Neuronal Profunda con Dropout sobre Fashion MNIST. 114
A.45 Tabla resumen de la Estructura de Red Neuronal Profunda con Dropout sobre Fashion MNIST. 114
A.46 Resultados de la Red Neuronal Profunda con Dropout sobre Fashion MNIST. . 115
A.47 Tabla de parámetros para la Red Neuronal Profunda con capa Agrupación Promedio Global sobre Fashion MNIST. 116
A.48 Tabla resumen de la Estructura de Red Neuronal Profunda con capa Agrupación Promedio Global sobre Fashion MNIST. 116

A.49 Resultados de la Red Neuronal Profunda con capa Agrupación Promedio Global sobre Fashion MNIST. 117
A.50 Tabla de parámetros para Red Neuronal Profunda con Normalización por Lotes sobre Fashion MNIST. 118
A.51 Tabla resumen de la Estructura de Red Neuronal Profunda con Normalización por Lotes sobre Fashion MNIST. 118
A.52 Resultados de la Red Neuronal Profunda con Normalización por Lotes sobre Fashion MNIST. 119
A.53 Tabla de parámetros para Red Neuronal Profunda con función de costo Similitud de Coseno sobre Fashion MNIST. 120
A.54 Tabla resumen de la Estructura de Red Neuronal Profunda con función de costo Similitud de Coseno sobre Fashion MNIST. 120
A.55 Resultados de la Red Neuronal Profunda con función de costo Similitud de Coseno sobre Fashion MNIST. 121
A.56 Tabla de parámetros para Red Neuronal Profunda con Convolución Dilatada sobre Fashion MNIST. 122
A.57 Tabla resumen de la Estructura de Red Neuronal Profunda con Convolución Dilatada sobre Fashion MNIST. 122
A.58 Resultados de la Red Neuronal Profunda con Convolución Dilatada sobre Fashion MNIST. 123
A.59 Tabla de parámetros para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre Fashion MNIST. 124
A.60 Tabla resumen de la Estructura de Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre Fashion MNIST. 124
A.61 Resultados de la Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre Fashion MNIST. 125
A.62 Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas sobre Fashion MNIST. 126

A.63 Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas sobre Fashion MNIST. 126
A.64 Resultados de la Red Neuronal Profunda con combinación de técnicas sobre Fashion MNIST. 127
A.65 Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre Fashion MNIST. 128
A.66 Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre Fashion MNIST. 129
A.67 Resultados de la Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre Fashion MNIST. . . . 129
A.68 Resultados del ensamblaje de todas las Redes Neuronales Profundas sobre Fashion MNIST. 130
A.69 Tabla de parámetros para Árbol de decisión sobre CIFAR-10. 131
A.70 Tabla de resultados de Árbol de decisión sobre CIFAR-10. 131
A.71 Tabla de parámetros para Bosque Aleatorio sobre CIFAR-10. 132
A.72 Tabla de resultados de Bosque Aleatorio sobre CIFAR-10. 132
A.73 Tabla de parámetros para Máquina de Soporte de Vectores sobre CIFAR-10. . . 132
A.74 Tabla de resultados de Máquina de Soporte de Vectores sobre CIFAR-10. . . . 133
A.75 Tabla de parámetros para Red Neuronal Profunda sobre CIFAR-10. 133
A.76 Tabla resumen de la Estructura de la Red Neuronal Profunda sobre CIFAR-10. 134
A.77 Resultados de la Red Neuronal Profunda sobre CIFAR-10. 134
A.78 Tabla de parámetros para Red Neuronal Profunda con Dropout sobre CIFAR-10. 135
A.79 Tabla resumen de la Estructura de la Red Neuronal Profunda con Dropout sobre CIFAR-10. 136
A.80 Resultados para Red Neuronal Profunda con Dropout sobre CIFAR-10. 136

A.81 Tabla de parámetros para la Red Neuronal Profunda con capa Agrupación Promedio Global sobre CIFAR-10. 137
A.82 Tabla resumen de la Estructura de Red Neuronal Profunda con capa Agrupación Promedio Global sobre CIFAR-10. 138
A.83 Resultados de la Red Neuronal Profunda con capa Agrupación Promedio Global sobre CIFAR-10. 138
A.84 Tabla de parámetros para Red Neuronal Profunda con Normalización por Lotes sobre CIFAR-10. 139
A.85 Tabla resumen de la Estructura de Red Neuronal Profunda con Normalización por Lotes sobre CIFAR-10. 140
A.86 Resultados de la Red Neuronal Profunda con Normalización por Lotes sobre CIFAR-10. 140
A.87 Tabla de parámetros para Red Neuronal Profunda con función de costo Similitud de Coseno sobre CIFAR-10. 141
A.88 Tabla resumen de la Estructura de Red Neuronal Profunda con función de costo Similitud de Coseno sobre CIFAR-10. 142
A.89 Resultados de la Red Neuronal Profunda con función de costo Similitud de Coseno sobre CIFAR-10. 142
A.90 Tabla de parámetros para Red Neuronal Profunda con Convolución Dilatada sobre CIFAR-10. 143
A.91 Tabla resumen de la Estructura de Red Neuronal Profunda con Convolución Dilatada sobre CIFAR-10. 144
A.92 Resultados de la Red Neuronal Profunda con Convolución Dilatada sobre CIFAR-10. 144
A.93 Tabla de parámetros para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre CIFAR-10. 145
A.94 Tabla resumen de la Estructura de Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre CIFAR-10. 146

A.95 Resultados de la Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre CIFAR-10. 146
A.96 Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas sobre CIFAR-10. 147
A.97 Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas sobre CIFAR-10. 148
A.98 Resultados de la Red Neuronal Profunda con combinación de técnicas sobre CIFAR-10. 148
A.99 Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento de la cantidad de datos sobre CIFAR-10. 150
A.100 Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento de la cantidad de datos sobre CIFAR-10. 150
A.101 Resultados de la Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento de la cantidad de datos sobre CIFAR-10. . . . 151
A.102 Resultados del ensamblaje de todas las Redes Neuronales Profundas sobre CIFAR-10. 152

Capítulo 1

Introducción

La Inteligencia Artificial (IA) es una disciplina científica que tiene como objetivo el desarrollo de sistemas computacionales (hardware/software) capaces de imitar la inteligencia humana o simular un comportamiento inteligente, no necesariamente humano. En la actualidad, la IA está presente en la mayoría de sistemas de software que gobiernan nuestra sociedad. Solo por dar algunos ejemplos, los asistentes virtuales de Google (Google Assistant) y Apple (Siri), las sugerencias de respuesta en Gmail e incluso los software encargados de clasificar los correos de tipo spam están impulsados por algoritmos de IA. Es más, de acuerdo a SITA[1] el 85 % de las aerolíneas planea implementar Chatbots para el año 2021 y el 79 % de los aeropuertos están usando actualmente IA para realizar análisis predictivos para mejorar su eficiencia operacional. También, en una encuesta a más de 630 compañías en 13 países se reporta que más del 30 % de las compañías han destinado $50M de dólares en iniciativas relacionadas con IA como la robótica, la automatización y el aprendizaje automático (Press, 2019).

Históricamente la IA se ha dividido en dos grandes paradigmas: simbólica y conexionista. La IA simbólica, paradigma dominante de la investigación en IA desde mediados de los cincuenta hasta finales de los años ochenta, trata de la representación del conocimiento desde un nivel más alto ”simbólico”(la lógica matemática y la resolución de problemas). Los nuevos desarrollos y la

[1] SITA es una compañía multinacional que provee de servicios informáticos a la industria de transporte aéreo, su página web es https://www.sita.aero/

investigación en IA introdujeron un enfoque basado en el comportamiento que emerge de redes formadas por unidades sencillas interconectadas. Hay muchas formas de conexionismo, pero la más común es el modelo de red neuronal.

En particular, la habilidad de los sistemas IA conexionistas para adquirir su propio conocimiento mediante la extracción de patrones de los datos se conoce como Aprendizaje Automático (AA) o Machine Learning (ML) (Goodfellow et al., 2016). El AA se ha utilizado para diferentes tareas, tales como la clasificación de imágenes en categorías, la transcripción de voz a texto, relacionar productos con los intereses de clientes, etc.

En los últimos años, la disponibilidad de recursos de hardware de alta capacidad de procesamiento (D'souza et al., 2020) y la generación de grandes cantidades de datos (Chen et al., 2013; Yang et al., 2019) ha producido un gran impulso en el desarrollo del Aprendizaje Profundo o Deep Learning (DL). En la Figura 1.1 se puede observar la relación entre los conceptos mencionados, entendiéndose que la Inteligencia Artificial engloba el Aprendizaje Automático y que el Aprendizaje Profundo es un subcampo del Aprendizaje Automático.

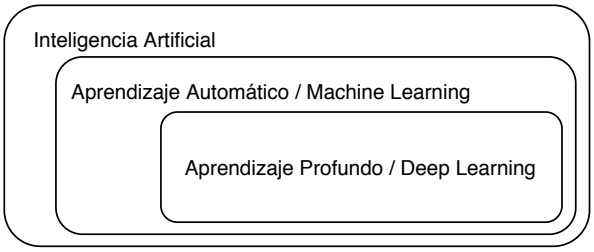

Figura 1.1: Diagrama conceptual de Aprendizaje Profundo (Goodfellow et al., 2016).

Los algoritmos de aprendizaje automático se pueden categorizar en tres tipos: **aprendizaje supervisado**, **aprendizaje no supervisado** y **aprendizaje por refuerzo**. El aprendizaje supervisado implica que el algoritmo aprende utilizando ejemplos de la tarea que debe realizar, por ejemplo, si la tarea es la clasificación de imágenes de animales en su respectiva categoría, entonces para su entrenamiento recibe ejemplos de animales con su respectiva categoría correctamente asignada. Este tipo de aprendizaje recibe su nombre de la noción de un instructor que le muestra los ejemplos y por lo tanto supervisa el entrenamiento. Por otro lado, el aprendizaje no supervisado implica que el algoritmo recibe los datos y debe obtener la relación entre ellos,

es decir, debe comprender de forma automática la estructura interna de los datos. Por último, el aprendizaje por refuerzo se caracteriza por poseer interacción entre un aprendiz y un entorno, donde este último se encarga de proveer de retroalimentación al primero. Específicamente, esta tesis se centra en los algoritmos de aprendizaje automático supervisado.

1.1. Problema

A pesar de los buenos resultados obtenidos por el aprendizaje automático supervisado, tanto clásico como profundo, existen algunas limitaciones, desafíos o problemas abiertos por resolver:

(a) **Etiquetado de una gran cantidad de datos.** Uno de los requisitos fundamentales para que los datos puedan utilizarse para generar buenos modelos de aprendizaje automático es que deben etiquetarse correctamente (ver Figura 1.2). Este requisito es una de las desventajas del Aprendizaje Profundo actual (Goodfellow et al., 2016) ya que el proceso de etiquetado puede resultar un proceso costoso y complejo que puede conllevar mucho tiempo y que variará dependiendo de la magnitud de los datos que se quieran etiquetar (Litjens et al., 2017).

Figura 1.2: Imágenes de animales correctamente etiquetadas.

(b) **Escasez o dificultad de recolección de datos.** Hay algunos campos que quedan fuera de esta tendencia de generación masiva de datos, entre ellos, el campo de la medicina (D'souza et al., 2020; Zhang et al., 2019) donde los datos son escasos, ya que están ligados a la cantidad de pacientes tratados, lo cual limita la cantidad de datos disponibles; otros campos fuera de esta generación masiva de datos son la industria (Le et al., 2020; Oviedo

et al., 2019) y la ciencia de materiales (Feng et al., 2019) donde existe una dificultad de recolección de datos debido a los procesos relacionados con su obtención (Barz y Denzler, 2019; Yang et al., 2019).

Por tanto, uno de los desafíos más importantes en la actualidad dentro del ámbito del aprendizaje automático (y del aprendizaje profundo) es la obtención de buenos modelos empleando conjuntos pequeños de datos.

1.1.1. Soluciones propuestas en la literatura

Para dar solución a los problemas antes planteados se han propuesto variadas técnicas en la literatura. En esta tesis, entendemos técnicas como la modificación o modificaciones que se puede realizar sobre un algoritmo de Aprendizaje Profundo para intentar mejorar su desempeño. Entre ellas se pueden destacar las siguientes: Aumento de Datos (Data Augmentation), la cual implica, para el caso de imágenes, aumentar la cantidad de datos que se tiene haciendo pequeñas modificaciones aleatorias, que pueden ser rotaciones, acercamientos o alejamientos, cambios de brillo, etc (Aurélien, 2019). También, está la Transferencia de Aprendizaje (Transfer Learning), la cual implica tomar modelos que ya han sido entrenados para un tipo de datos y utilizar el conocimiento que han obtenido para aplicarlo sobre otro conjunto de datos que sea de un dominio cercano. Otra técnica utilizada es la Regularización la cual es un conjunto de técnicas cuyo objetivo es mitigar el error que obtiene un modelo durante su etapa de evaluación (Goodfellow et al., 2016), es decir, cuando es puesto a prueba con datos que no están contenidos dentro de los ejemplos de entrenamiento. Además de las técnicas mencionadas, hay otras más, las cuales se definen y describen en el Capítulo 4.

1.2. Hipótesis y objetivos

1.2.1. Hipótesis

En base a lo anterior se plantea la siguiente hipótesis: es posible determinar cuál es la técnica o conjunto de técnicas más adecuado para clasificar imágenes empleando aprendizaje profundo

cuando se cuenta con un conjunto pequeño de datos de entrenamiento. La confirmación de esta hipótesis podría tener implicaciones en la disponibilidad de los algoritmos de aprendizaje automático profundo a universidades, empresas y personas naturales que cuenten con recursos (computacionales, monetarios o de tiempo) limitados.

1.2.2. Objetivos

Esta tesis tiene como objetivo comparar el efecto de cada técnica sobre el desempeño general del algoritmo de aprendizaje para encontrar la técnica o combinación de ellas que permita reducir el impacto de tener un conjunto pequeño de datos de entrenamiento. Para ello se identifican las técnicas propuestas en la literatura, se realiza una implementación de cada una de ellas, se combinan técnicas y se comparan resultados. Además, se agregan algoritmos de aprendizaje clásicos, tales como Árboles de Decisión, Bosque Aleatorio y Máquina de Soporte de Vectores para contrastar los resultados obtenidos con los algoritmos de Aprendizaje Profundo y las distintas técnicas. Si bien no hay un consenso general de que tamaño significa conjunto pequeño, para esta tesis se considera un conjunto pequeño de datos aquel conjunto que tenga entre 10 y 500 ejemplos por clase.

1.2.2.1. Objetivos específicos

(a) Desarrollar, analizar y evaluar los algoritmos de aprendizaje clásicos, luego anotar resultados obtenidos sobre conjuntos pequeños de datos.

(b) Desarrollar, analizar y evaluar redes neuronales profundas sin técnicas adicionales, luego anotar resultados obtenidos sobre conjuntos pequeños de datos.

(c) Implementar algoritmos de redes neuronales profundas con las diferentes técnicas de la literatura, luego anotar resultados obtenidos sobre conjuntos pequeños de datos.

(d) Combinar las diferentes técnicas implementadas y evaluar el desempeño de las mismas.

(e) Presentar cuadro comparativo de los resultados obtenidos por los algoritmos.

(f) Presentar conclusiones sobre la hipótesis.

1.3. Organización de la tesis

La organización de la tesis es la siguiente:

- En el Capítulo 2 se define el Aprendizaje Automático y sus componentes, se ejemplifica su proceso de entrenamiento y describen las métricas utilizadas para su evaluación.
- En el Capítulo 3 se introducen y ejemplifica el funcionamiento de los algoritmos de Aprendizaje Automático clásicos, tales como los Árboles de Decisión, Bosque Aleatorio y Máquina de Soporte de Vectores. También, se presenta el Aprendizaje Profundo, es decir, las Redes Neuronales Profundas y las capas que le componen. Además, se presenta el problema de utilizar estos métodos cuando se cuenta con un conjunto pequeño de datos.
- En el Capítulo 4 se describen las técnicas de Aprendizaje Profundo que han sido propuestas en la literatura para intentar mejorar el desempeño de los algoritmos cuando se enfrentan a un conjunto pequeño de datos. Se presenta una tabla comparativa entre los trabajos estudiados.
- En el Capitulo 5 se describen las técnicas de Aprendizaje Profundo que han sido propuestas en la literatura para intentar mejorar el desempeño de los algoritmos cuando se realiza un entrenamiento con un conjunto pequeño de datos y se presenta una tabla comparativa entre los trabajos estudiados.
- En el Capítulo 6 se detallan los experimentos realizados junto con sus aspectos técnicos y presentan los resultados de los mismos. Luego, se presenta una discusión sobre los resultados obtenidos.
- Por último, se presentan los aportes de la tesis, las conclusiones del trabajo realizado en los experimentos y los resultados que estos entregan. Y, además, se mencionan algunas líneas de trabajo futuro.

Capítulo 2

Aprendizaje automático

Desde el punto de vista del campo de la computación, el aprendizaje automático es la ciencia de la programación de algoritmos que pueden aprender de los datos (Aurélien, 2019), este aprendizaje se obtiene mediante la extracción de patrones de los datos (Goodfellow et al., 2016). Por ejemplo, Tom Mitchell en su libro Machine Learning (Mitchell, 1997) da la siguiente definición:

> *"Se dice que un algoritmo* ***aprende*** *de la experiencia E respecto de una clase de tareas T y una medida de desempeño P, si su desempeño en la tarea T, medida por P, mejora con la experiencia E".*

La tarea T es el objetivo del algoritmo. La tarea se describe en términos de cómo se debe procesar un ejemplo. Este ejemplo debe ser una medición cuantitativa de un objeto que se quiere que el algoritmo procese, es decir, debe estar expresado en valores numéricos. Un ejemplo se representa como un vector $x \in \mathbb{R}^n$ donde cada x_i es una característica del ejemplo. Por ejemplo, una imagen se representa con un vector donde cada elemento es un píxel de la imagen. Una de las tareas es la clasificación. El objetivo de esta tarea es determinar a cuál de k categorías corresponde una entrada. Para ello, se le pide al algoritmo de aprendizaje que produzca una función $f : \mathbb{R}^n \to \{1, \ldots, k\}$, cuya salida es la codificación numérica de las categorías y la entrada es la representación vectorial de la entrada. La función que se obtiene se denomina modelo.

La medida P, de la palabra en inglés *Performance* que significa desempeño, es la métrica que se utiliza para evaluar el desempeño del algoritmo. Para la clasificación se emplea la medida **exactitud** del modelo. La exactitud es la proporción de entradas para las cuales el modelo entrega la clasificación correcta, respecto de la cantidad de entradas totales. También se utiliza el **error** del modelo, el cual es la proporción de entradas para las cuales el modelo entrega la clasificación incorrecta.

La experiencia E hace referencia a los ejemplos que se utilizan para que el algoritmo aprenda. Un **conjunto de datos** es una colección de ejemplos utilizados para que un algoritmo aprenda. Es aquí, en los tipos de conjuntos de datos que utilizan, donde los tipos de aprendizaje se diferencian. Como hemos mencionado anteriormente, los algoritmos de aprendizaje no supervisado utilizan conjuntos de datos con muchas características y aprenden las propiedades de la estructura que presentan dichos datos. Mientras que los algoritmos de aprendizaje supervisado utilizan conjuntos de datos con características identificadas previamente por una **etiqueta** asociada a cada dato. Con esto el algoritmo es capaz de aprender la relación entre los datos y las etiquetas y de relacionar datos nuevos.

2.1. Entrenamiento

El entrenamiento de un algoritmo de aprendizaje automático se puede ilustrar por una regresión lineal. En una regresión lineal se intenta aproximar la función $\hat{y}(x)$ a la función $y(x)$ donde $x \in \mathbb{R}^n$ es el vector de entrada:

$$\hat{y} = w^T x \tag{2.1}$$

donde $w^T \in \mathbb{R}^n$ es un vector de **parámetros** transpuesto.

Los parámetros son variables que controlan el comportamiento del sistema. Para este caso cada w_i es un coeficiente que multiplica a cada componente x_i. Se puede pensar en w como los pesos de cada característica de x que determinan cuánto afecta cada característica a la predicción. Si la característica x_i recibe un peso positivo w_i, entonces al incrementar dicha característica aumentará la predicción $\hat{y}$. Al contrario, si dicha característica recibe un peso negativo, entonces

al incrementar el valor de la característica obtendremos una reducción de la predicción. Por lo tanto, si la magnitud del peso es grande, entonces el impacto de la característica en la predicción es grande.

Para evaluar el desempeño de la aproximación se utiliza la métrica de error cuadrático medio (MSE, por sus siglas en inglés). En la siguiente ecuación se muestra como se calcula esta métrica para un conjunto de datos que tiene n ejemplos:

$$MSE = \frac{1}{n}\sum_{i}^{n}(\hat{y} - y)_i^2 \tag{2.2}$$

El entrenamiento se realiza sobre un conjunto de datos llamado **conjunto de entrenamiento**, sobre el cual se calcula el error cuadrático medio de entrenamiento o **error de entrenamiento** con la Ecuación 2.2.

Para optimizar el proceso de entrenamiento se intenta minimizar el error resolviendo para un gradiente de 0:

$$\nabla_w MSE_{(}entrenamiento) = 0 \tag{2.3}$$

Cuando el proceso de entrenamiento termina se utiliza un conjunto de datos llamado **conjunto de prueba** para calcular un error cuadrático medio de prueba o **error de prueba**. La idea central de este conjunto de prueba es que tenga ejemplos nuevos, que no estén en el conjunto de entrenamiento, para poder evaluar la habilidad de generalización del entrenamiento. Por ejemplo, para un conjunto de prueba con m ejemplos el error de prueba se calcula de la siguiente manera:

$$MSE_{(}prueba) = \frac{1}{m}\sum_{i}(\hat{y} - y)_i^2 \tag{2.4}$$

La función que se utiliza para calcular el error, ya sea de entrenamiento o de prueba, se denomina **función de costo**. Para el ejemplo se utilizó el error cuadrático medio, pero también se podría utilizar una métrica alternativa. Por ejemplo, para el caso de una clasificación se puede utilizar el número de imágenes correctamente clasificadas.

Bajo ajuste y sobreajuste. El principal objetivo de un algoritmo de aprendizaje automático es, además de resolver la tarea de aprendizaje, desempeñarse bien sobre entradas que no se encuentran en el conjunto de entrenamiento. Los factores que determinan el desempeño un algoritmo de aprendizaje profundo son su habilidad de:

(a) Reducir el error de entrenamiento.

(b) Reducir la diferencia entre el error de entrenamiento y el error de prueba.

Estos factores corresponden a dos problemas que se enfrentan durante el aprendizaje: **underfitting** o bajo ajuste y **overfitting** o sobreajuste. El primero ocurre cuando el modelo no es capaz de reducir el error de entrenamiento. El último ocurre cuando la diferencia entre el error de entrenamiento y el error de prueba es muy grande (Goodfellow et al., 2016).

Para ejemplificar se muestra la Figura 2.1, donde se ha representado en una gráfica los puntos de una función que se comporta de manera cuadrática. Sin embargo, el modelo que presenta bajo ajuste aproxima una función lineal y el modelo que presenta sobreajuste aproxima una función polinomial. El ideal se representa por el modelo de en medio, que presenta un ajuste correcto.

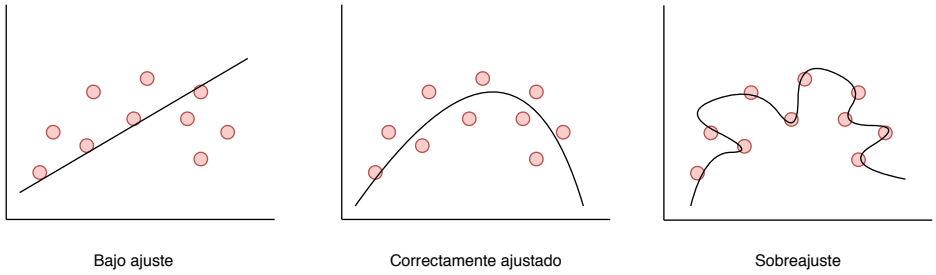

Figura 2.1: Ejemplo de desempeño bajo ajuste, correctamente ajustado y sobreajuste.

2.2. Métricas para clasificación

En la tarea de clasificación se pide al algoritmo de aprendizaje que identifique a cuál de k categorías corresponde una entrada (Goodfellow et al., 2016). Estas categorías son también denominadas clases.

Para la definición de las métricas de clasificación se utiliza la matriz de confusión, ver Figura 2.2. Las filas de la matriz representan la clase predicha y las columnas representan el valor real

de la clase. Esta sección está basada en Hossin y Sulaiman (2015).

		Real	
		Positivo	Negativo
Predicción	Positivo	Verdadero Positivo (VP)	Falso Positivo (FP)
	Negativo	Falso Negativo (FN)	Verdadero Negativo (VN)

Figura 2.2: Matriz de confusión.

De la matriz se obtienen los siguientes valores, VP y VN son las cantidad de predicciones realizadas de manera correcta. FP y FN son las predicciones que el clasificador hizo de manera incorrecta.

Exactitud o Accuracy. La Exactitud o Accuracy es la razón entre la cantidad de clasificaciones correctas y el total de clasificaciones realizadas. Para un problema de clasificación de dos clases, se define como:

$$Exactitud = \frac{VP + VN}{VP + FP + VN + FN} \tag{2.5}$$

Retorna un valor entre 0 y 1, siendo 1 su mejor valor.

Precisión. La precisión es la razón entre la cantidad de clasificaciones correctas de una clase positiva y la cantidad de clasificaciones totales de una clase positiva. La ecuación que la define en función de la matriz de confusión es:

$$Precisión = \frac{VP}{VP + FP} \tag{2.6}$$

Retorna un valor entre 0 y 1, siendo 1 su mejor valor.

Recall. Recall es la razón entre la cantidad de clasificaciones correctas de una clase positiva y la cantidad de clasificaciones correctas. La ecuación que la define en función de la matriz de

confusión es:

$$Recall = \frac{VP}{VP + VN} \tag{2.7}$$

Retorna un valor entre 0 y 1, siendo 1 su mejor valor.

F1. F1 es la media armónica entre recall y precisión. La media armónica se denota por la letra H y se calcula con la siguiente ecuación:

$$H = \frac{n}{(\frac{1}{x_1} + \frac{1}{x_2} + \cdots + \frac{1}{x_n})} \tag{2.8}$$

Si $x_1 = Precisión$ y $x_2 = recall$ entonces F1, se define como:

$$F1 = \frac{2}{\frac{1}{precisión} + \frac{1}{recall}} \tag{2.9}$$

$$F1 = \frac{2}{\frac{1}{\frac{recall + precisión}{precisión \cdot recall}}} \tag{2.10}$$

$$F1 = 2 \cdot \frac{Precisión \cdot Recall}{Precisión + Recall} \tag{2.11}$$

Retorna un valor entre 0 y 1, siendo 1 su mejor valor.

Coeficiente de correlación de Matthews. La exactitud y el F1 pueden generar resultados confiables cuando el conjunto de datos es balanceado, es decir la cantidad de ejemplos por clase es la misma para todas las clases, pero si este no fuera el caso, los resultados pueden ser engañosos debido a que fallan al considerar la razón entre elementos positivos y negativos Chicco y Jurman (2020).

Dado este problema se añade el Coeficiente de Correlación de Matthews (1975) (MCC), el cual es considerado un indicador del desempeño general del algoritmo, ya que considera todos los elementos positivos y negativos, es decir, entrega resultados confiables incluso con conjuntos

de datos no balanceados. La ecuación para MCC en función de la matriz de confusión es:

$$MCC = \frac{VP \cdot VN - FP \cdot FN}{\sqrt{(VP + FP)(VP + FN)(VN + FP)(VN + FN)}} \tag{2.12}$$

Retorna un valor entre -1 y +1, donde +1 representa predicción perfecta, 0 representa predicción aleatoria y -1 representa predicción totalmente errónea.

2.3. Funciones de costo

La función de costo calcula el error en términos numéricos entre la predicción y la verdad fundamental, por ejemplo, para una regresión, la Ecuación 2.2 calcula la distancia entre un punto predicho y la ubicación verdadera de dicho punto. También, se puede entender la función de costo como el *precio* que se está dispuesto a pagar por predecir utilizando un modelo (Rosasco et al., 2004), siendo el objetivo último obtener una predicción que tiene precio cero, es decir, sin error entre lo que se predice y la verdad. La función de costo en un algoritmo de Aprendizaje Automático usualmente se reduce a la suma de una función que se calcula para cada ejemplo (Goodfellow et al., 2016).

2.3.1. Entropía cruzada Categórica.

La función de costo Entropía cruzada Categórica, o en inglés, Categorical Cross-Entropy (CCE) se compone de dos funciones llamadas Softmax y Entropía Cruzada.

Codificación One-hot. La Codificación One-hot es una forma de representar las clases, la cual consiste en utilizar un vector de largo equivalente a la cantidad de clases que hay, donde la posición del vector que representa una clase tiene el valor 1 y el resto 0. Por ejemplo en la Figura 2.3 se muestra la representación en formato one-hot para tres clases.

Etiqueta	Representación One-hot
Pingüino	[1 0 0]
Perro	[0 1 0]
Gato	[0 0 1]

Figura 2.3: Ejemplo de Codificación One-hot para tres clases.

Función Softmax. La función Softmax convierte un vector de K valores en un vector de K valores que suman 1. Los valores del vector de entrada pueden ser positivos, negativos o cero. La función Softmax convierte los valores de entrada al rango [0,1]. El vector de salida se puede como un vector de probabilidades (Wood, s.f.). La función se define:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{2.13}$$

Donde todos los valores z_i son los elementos del vector de entrada. El denominador de la fracción es la sumatoria que genera la normalización de los valores para que se produzca el rango [0,1] (Wood, s.f.).

Entropía Cruzada. La función de costo Entropía Cruzada, o en inglés, Cross-Entropy aumenta a medida que el valor predicho se aleja del valor verdadero. Por ejemplo predecir 0,30 cuando cuando el valor real es 1 produce un valor de costo alto. Por otro lado, si el modelo fuera perfecto el valor de costo será 0, cuando el valor predicho es igual al valor verdadero (Gómez, 2018). La función se define como:

$$CE = -\sum_{i}^{K} t_i log(s_i) \tag{2.14}$$

Donde t_i es el valor verdadero para la clase i y s_i es el valor predicho por el modelo para la clase i (Gómez, 2018). El vector que contiene los valores verdaderos debe estar en representación One-Hot.

Función de costo Entropía cruzada Categórica. La función de costo Entropía cruzada Categórica se compone de las funciones antes descritas. En la Figura 2.4 se observa un diagrama que muestra la composición de la función CCE y el flujo de la información que calcula.

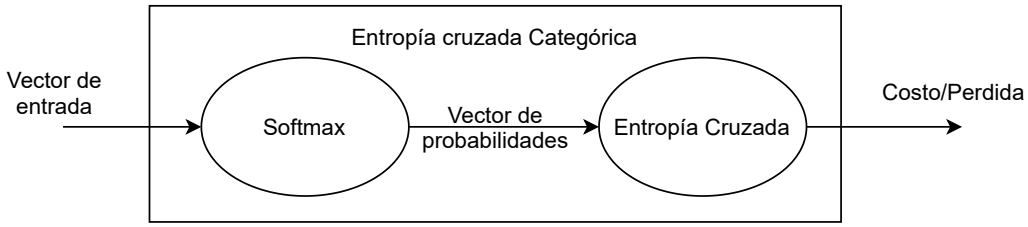

Figura 2.4: Diagrama de la composición de la función de costo Entropía cruzada Categórica.

De esta manera la función de costo Entropía cruzada Categórica mide el costo total de las predicciones que hace el clasificador para K clases. Gracias a la función Entropía Cruzada, la función Entropía cruzada Categórica asigna un costo alto a un error alto y un costo bajo a un error bajo. Para un algoritmo de predicción perfecto, esta función de costo entregaría un valor 0.

Capítulo 3

Algoritmos de aprendizaje automático

3.1. Árbol de decisión

Los Árboles de Decisión son algoritmos de Aprendizaje Automático versátiles capaces de realizar tanto la tarea de clasificación como la tarea de regresión. Son capaces de ajustarse correctamente a conjuntos de datos que tienen alta complejidad (Aurélien, 2019).

Un Árbol de decisión está compuesto por nodos, comenzando por el nodo raíz que no tiene conexiones entrantes. Los demás nodos tienen solo una conexión entrante. Un nodo con conexiones salientes se denomina nodo interno. Un nodo sin conexiones salientes se denomina nodo terminal o nodo hoja. En un Árbol de decisión cada nodo interno divide los datos en dos o más partes basándose en el valor de uno o más atributos. En cada nodo hoja se asigna una clase en las cuales se quiere clasificar los datos. La clasificación se realiza mediante la navegación del árbol desde la raíz hasta las hojas según los nodos intermedios indiquen (Rokach y Maimon, 2006).

Por ejemplo, en la Figura 3.1, se muestra un Árbol de decisión capaz de asignar una cantidad de descuento a una solicitud de compra basado en la cantidad de productos y la dirección de envío de la solicitud.

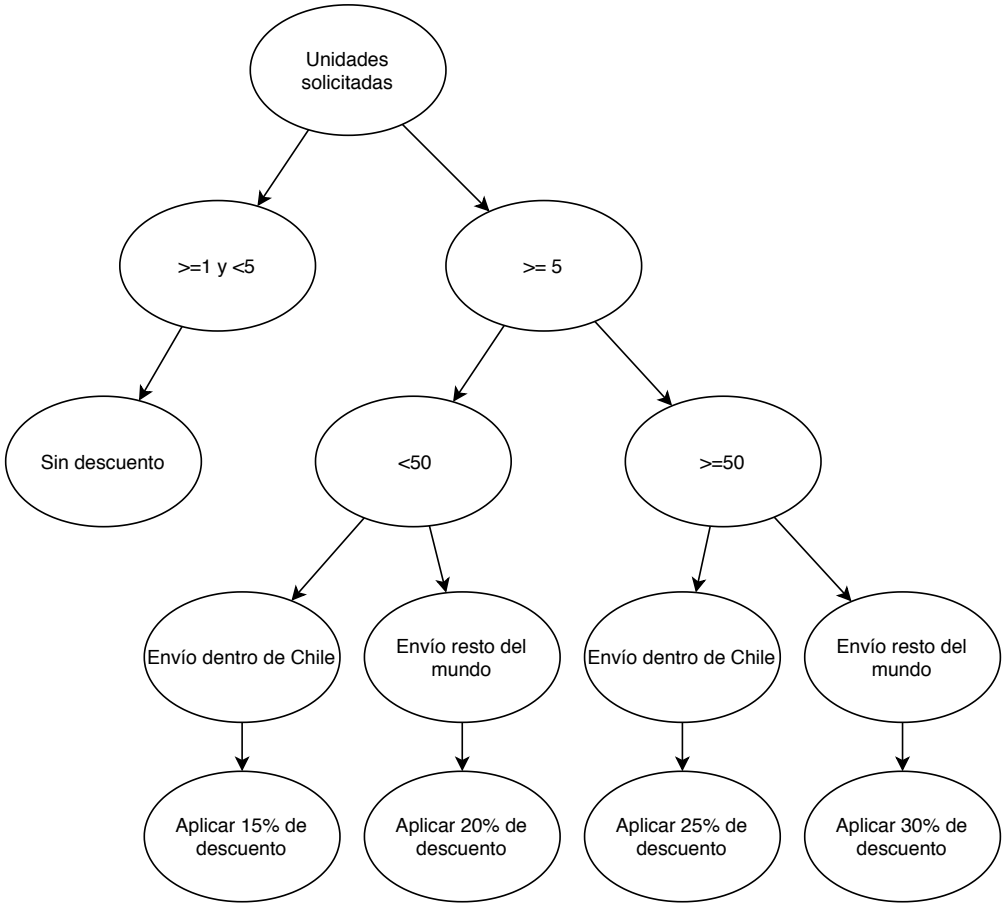

Figura 3.1: Ejemplo de Árbol de decisión.

De la Figura 3.1 se puede observar que las decisiones que toma el modelo pueden ser comprendidas de manera intuitiva mediante el seguimiento de los caminos desde la raíz hasta las hojas.

3.1.1. Bosque aleatorio

Los Árboles de Decisión son los componentes fundamentales de los Bosques Aleatorios que son considerados uno de los algoritmos de Aprendizaje Automático más poderosos hoy en día (Aurélien, 2019).

El algoritmo de Bosque Aleatorio, o en inglés, *Random Forest* es un clasificador consistente en

un conjunto de clasificadores de tipo árbol de decisión independientes donde, dada una entrada, cada árbol emite un voto unitario sobre la clase que considera que corresponde a dicha entrada, luego el Bosque Aleatorio tiene como salida (predicción) la clase más votada por los árboles internos (Breiman, 2001). Un diagrama de esto se puede observar en la Figura 3.2.

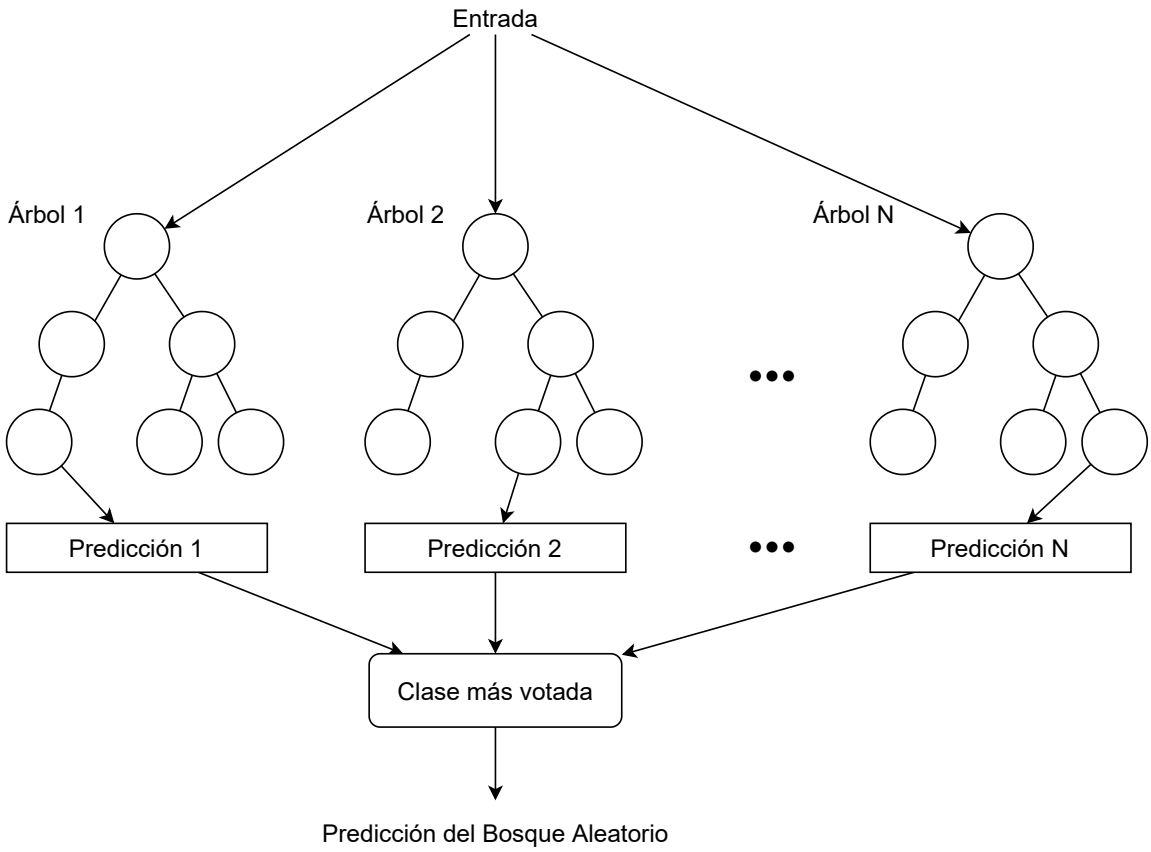

Figura 3.2: Diagrama de predicción por Bosque Aleatorio.

3.2. Máquina de soporte de vectores

Una Máquina de Soporte de Vectores (Cortes y Vapnik, 1995), o en inglés Support Vector Machine (SVM), es un modelo de Aprendizaje Automático capaz de realizar regresiones y clasificaciones tanto lineales como no lineales (Aurélien, 2019).

La idea de la SVM se puede observar en la Figura 3.3 donde hay dos clases separables por

una línea recta, es decir,son linealmente separables.

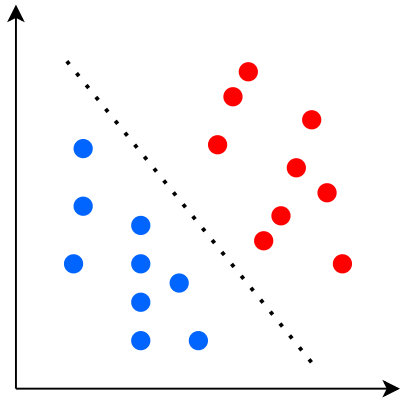

Figura 3.3: Separación de datos por SVM.

Se puede pensar en la SVM como un clasificador que busca encontrar la separación más grande entre las dos clases, representada por las dos líneas punteadas en la Figura 3.4. Esta clasificación se denomina *clasificación de gran margen* (Aurélien, 2019).

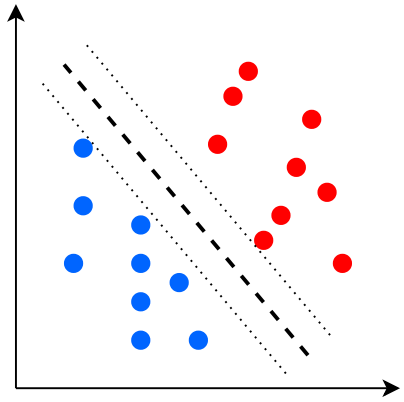

Figura 3.4: Separación de datos con gran margen.

Aunque una SVM lineal funciona bien para muchos casos, hay casos en que los datos no son linealmente separables (Aurélien, 2019). Ahí es donde, para conjuntos de datos más interrelacionados, se utiliza una técnica llamada "Kernel Trick"que permite aumentar la dimensionalidad de los datos. Esto permite la separación de datos que se encuentran mezclados en una determinada

dimensión. Por ejemplo en la Figura 3.5 se llevan los datos de dos dimensiones a tres dimensiones, permitiendo la separación de estos.

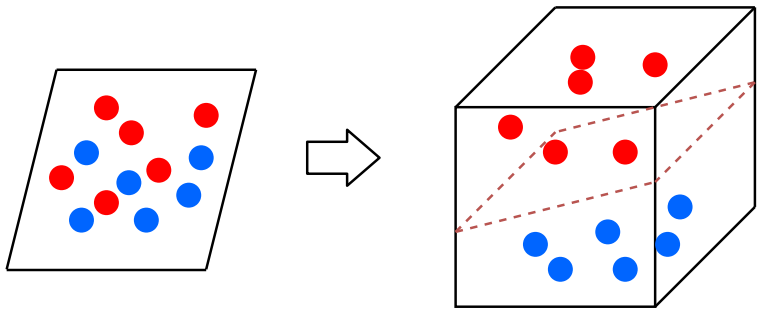

Figura 3.5: Separación de datos altamente interrelacionados utilizando kernel trick.

3.3. Redes neuronales artificiales

Las Redes Neuronales Artificiales son modelos computacionales inspirados en las redes neuronales biológicas que se encuentran en el cerebro (Chen et al., 2019). Estas redes neuronales están compuestas por neuronas artificiales llamadas perceptrones. En la Figura 3.6 se observa un modelo del perceptrón.

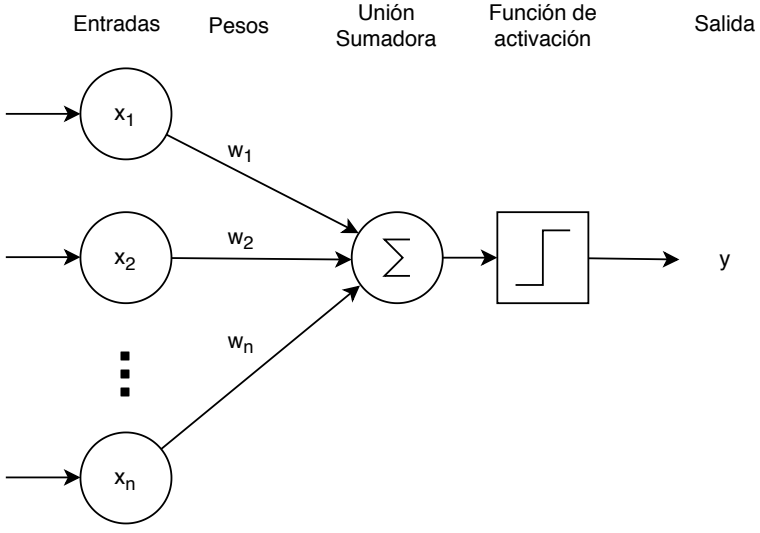

Figura 3.6: Modelo de perceptrón propuesto por Rosenblatt (1958).

El perceptrón es diseñado para ilustrar las propiedades fundamentales de los sistemas inteligentes, pero sin adentrarse demasiado en las condiciones, a veces desconocidas, de los sistemas biológicos (Rosenblatt, 1958).

Como se observa en la Figura 3.6, el perceptrón se compone de cinco elementos. De izquierda a derecha, las señales de entrada, los pesos de cada entrada, una unión sumadora, una función de activación y la salida. La salida y del perceptrón se calcula con la Ecuación 3.1:

$$y = f(\sum_{i=1}^{n} x_i \cdot w_i) \tag{3.1}$$

Donde f es la función de activación. Esta función se utiliza como una forma de fijar un nivel mínimo de actividad a la neurona para considerarse como activa. Es decir, obtiene una sumatoria de entradas suficientemente relevante como para ser considerada válida. En la literatura se utilizan varias de estas funciones (Nwankpa et al., 2018), las más importantes son las siguientes:

- Sigmoide:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.2}$$

- Tangente Hiperbólica:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3.3}$$

- Softmax:

$$f(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{3.4}$$

- Unidad lineal rectificada (ReLU, por sus siglas en inglés):

$$f(x) = max(0, x) \tag{3.5}$$

De las funciones de activación expuestas arriba, la más utilizada es ReLU debido a que garantiza un tiempo de cómputo menor ya que no calcula exponentes ni divisiones (Nwankpa et al., 2018).

3.3.1. Perceptrón multicapa

El perceptrón multicapa es también conocido como Red Neuronal Prealimentada o, en inglés, Feedforward Neural Network. El perceptrón multicapa consiste de tres capas, una primera capa de entrada, una segunda capa oculta y una tercera capa de salida. El modelo que describe el perceptrón multicapa se presenta en la Figura 3.7:

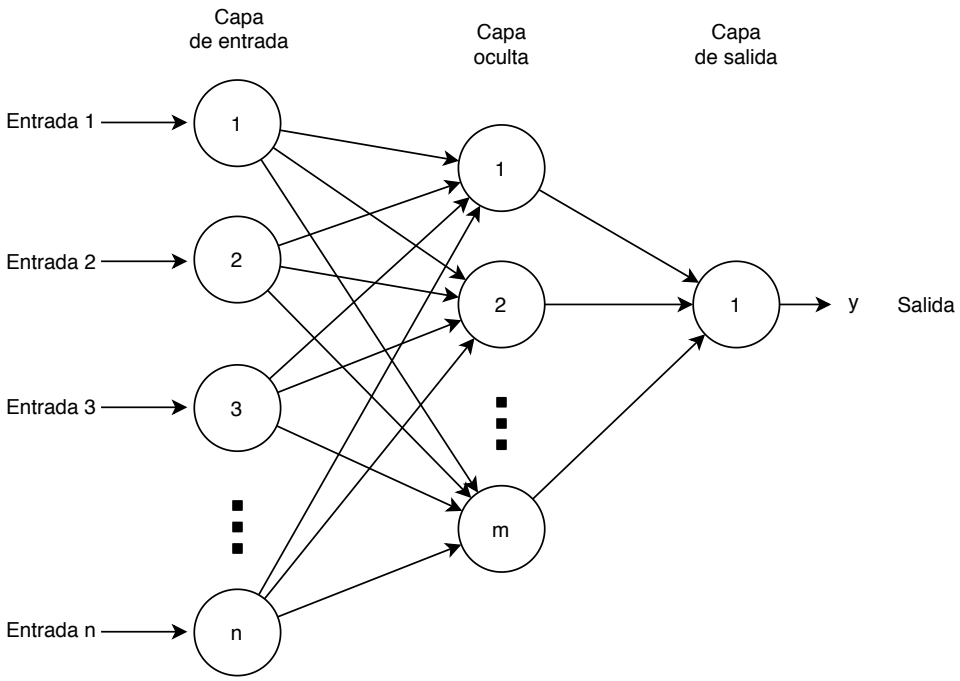

Figura 3.7: Modelo del perceptrón multicapa o Red Neuronal Prealimentada (Tang y Kwan, 1993).

En cada capa del modelo de la Figura 3.7 se calcula la salida para cada neurona utilizando la Ecuación 3.1 con una función de activación previamente definida.

3.4. Redes neuronales profundas

Según lo establecido en la sección anterior, las Redes Neuronales Profundas son Redes Neuronales Prealimentadas (Feedforward Neural Networks) que tienen múltiples capas ocultas. Un ejemplo con dos capas ocultas se puede observar en la Figura 3.8.

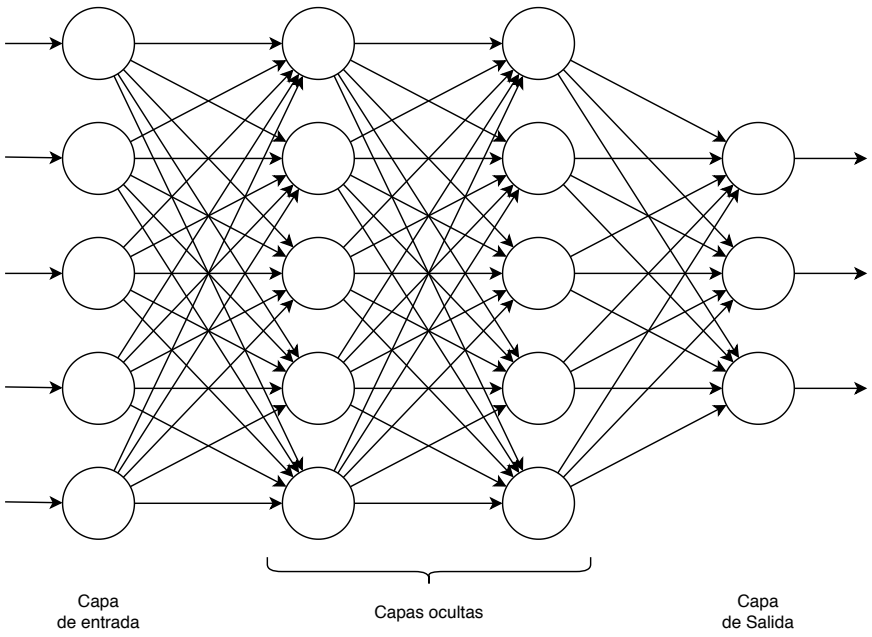

Figura 3.8: Ejemplo de Red Neuronal Profunda con dos capas ocultas.

El entrenamiento de una red neuronal consiste en ajustar los pesos de las neuronas usando un algoritmo llamado Retro-propagación o, en inglés, Back-Propagation (Rumelhart et al., 1986). Por ejemplo, dada una red neuronal cuyos pesos han sido inicializados con valores positivos muy pequeños, se utiliza la función de costo para calcular el error entre los valores que esta red predice y los valores reales. El objetivo del entrenamiento es reducir el error de la predicción de la red neuronal, por lo tanto, se calcula el gradiente negativo de la función de costo, el cual indica al algoritmo de Retro-propagación cuanto debe ajustar los pesos para reducir el error. Para calcular el gradiente negativo se utiliza el algoritmo de Descenso del Gradiente (Gradient Descent). Lamentablemente, este algoritmo es computacionalmente costoso cuando se necesita calcular con una cantidad de datos relativamente grande.

Una modificación del algoritmo de Descenso del Gradiente, llamado Descenso del Gradiente Estocástico es el algoritmo más utilizado para realizar entrenamientos de redes neuronales (Goodfellow et al., 2016), ya que ofrece una menor carga computacional. Un parámetro crucial para este algoritmo es la Tasa de Aprendizaje o, en inglés, *learning rate*, ya que indica la velocidad de convergencia, por lo que un valor muy pequeño podría hacer que el algoritmo tarde

mucho en converger y un valor muy alto podría hacer que nunca logre converger. Durante el entrenamiento de una Red Neuronal Profunda este algoritmo se utiliza en lotes por lo que se define un Tamaño de Lote, o en inglés, Batch Size que indica la cantidad de ejemplos a los que se calculará su gradiente.

El proceso de entrenamiento se lleva a cabo por una cantidad de iteraciones determinada al inicio del proceso, esta cantidad se denomina épocas o iteraciones de entrenamiento. También, se puede utilizar la Detención Temprana, o en inglés, Early Stopping, que permite monitorear una métrica a elección, y dado el comportamiento de dicha métrica durante el entrenamiento, se puede detener el proceso antes de alcanzar la cantidad de épocas establecida. Para lograr lo anterior, se introduce un parámetro llamado Paciencia, el cual define la cantidad de iteraciones de entrenamiento que se va a esperar antes de detener el proceso.

A continuación se presentan las capas que se utilizan en los algoritmos de Aprendizaje Profundo que se implementan en esta tesis. Es importante destacar que el orden y la cantidad de las capas de una red neuronal se denomina arquitectura de la red. Estas arquitecturas se pueden construir con las siguientes capas: la capa densa, capa convolucional, capa de agrupación, entre otras.

3.4.1. Capa densa

La capa densa es una capa de una red neuronal que se compone de n perceptrones, también denominados unidades. Su principal característica es que cada perceptrón está conectado con todas las entradas de la siguiente capa. Un ejemplo de esta capa se observa en la Figura 3.8. A esta capa también se le denomina totalmente conectada o *fully connected*.

3.4.2. Capa convolucional

La Capa Convolucional utiliza una operación llamada Convolución. La Convolución es una operación matemática que desliza una función sobre otra y mide el integral de su multiplicación

(Aurélien, 2019). Se define matemáticamente como:

$$s(t) = \int x(a)w(t-a)da \tag{3.6}$$

La operación convolución se denota con un asterisco (Goodfellow et al., 2016):

$$s(t) = (x * w)(t) \tag{3.7}$$

Para ejemplificar el funcionamiento de las capas convolucionales se utiliza como entrada una imagen, representada por una matriz de dos dimensiones, donde cada celda contiene el valor de un píxel. Las neuronas en la primera capa convolucional no están conectadas a cada píxel de la imagen de entrada sino que están conectadas sólo con los píxeles que están en sus campos de recepción (Aurélien, 2019). En la Figura 3.9 se puede observar esto, donde, por ejemplo, en la primera subfigura de la izquierda, la neurona de la izquierda (más oscura) sólo está conectada con los nueve píxeles de la izquierda (más oscuros). Para las siguientes capas convolucionales se repite la misma idea, pero con la salida de la capa que le precede.

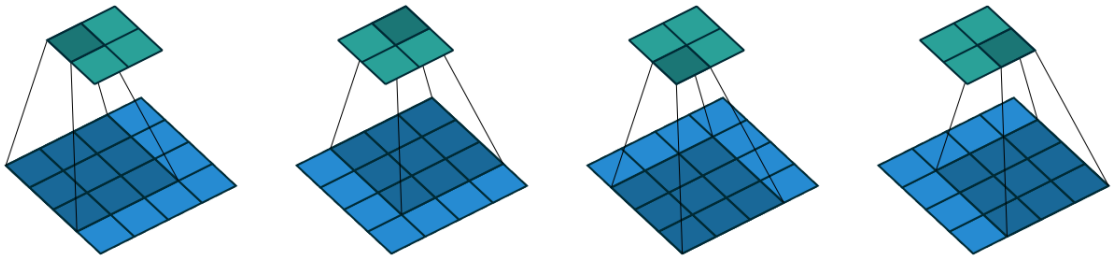

Figura 3.9: Ejemplo de operación de convolución. Imagen tomada de Dumoulin y Visin (2016).

El Campo receptivo es la cantidad de píxeles a los cuales está conectada cada neurona. Para la Figura 3.9 este toma el valor de 3x3. La cantidad de píxeles que se desplaza cada neurona es llamada paso (stride), y para la Figura 3.9 es de 1.

Para que una capa tenga el mismo alto y ancho que la capa que le precede es común agregar ceros alrededor de la entrada, como se puede ver en la Figura 3.10. Esto se conoce como *zero*

padding (Aurélien, 2019).

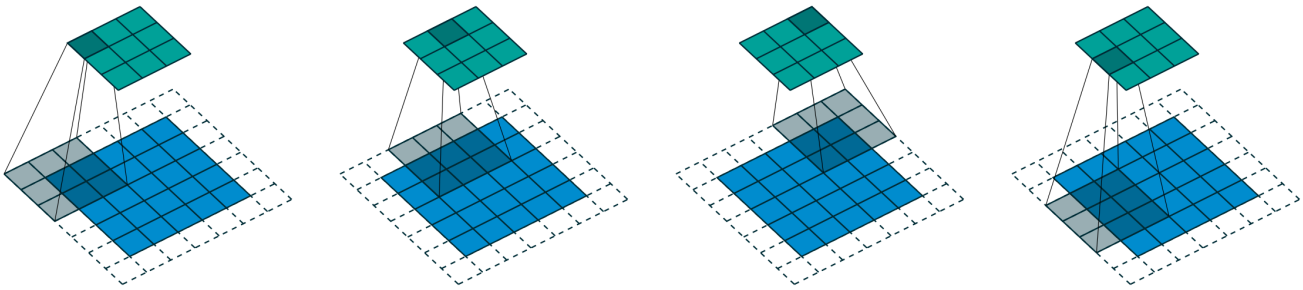

Figura 3.10: Ejemplo de operación de convolución con padding. Imagen tomada de Dumoulin y Visin (2016).

Para la Figura 3.10 el campo receptivo sigue siendo 3x3, pero su paso es de 2.

Los pesos de cada neurona se representan como una imagen pequeña del tamaño del campo receptivo. Los pesos se denominan filtros o kernels. Estos pesos se ajustan durante el periodo de entrenamiento formando patrones dependiendo de la tarea que se le encargue aprender. Un ejemplo del cálculo de la salida con un filtro se puede observar en la Figura 3.11

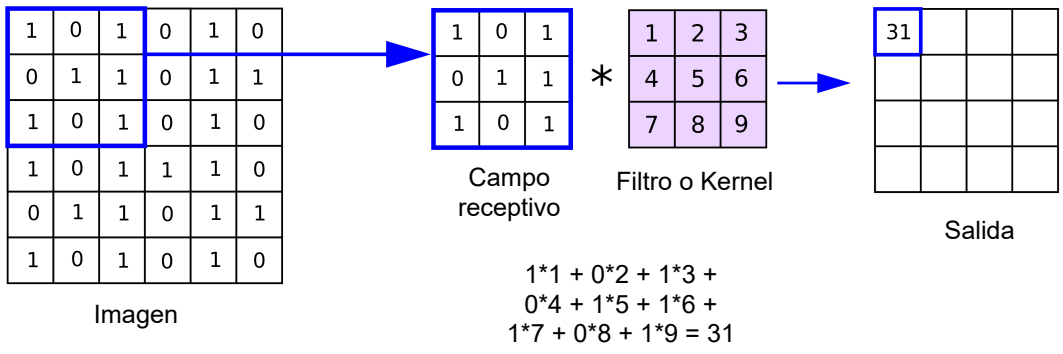

Figura 3.11: Operación de convolución. Imagen tomada de Reynolds (2019).

3.4.3. Capa aplanar

La capa de Aplanar o, en inglés, Flatten Layer es una capa que permite conectar una capa convolucional a una capa densa mediante la transformación de su entrada en un vector unidi-

mensional. Es decir, recibe una entrada en dos dimensiones, la cual es transformada por la capa Aplanar a un vector de una dimensión, el cual es luego entregado a una capa densa, permitiendo así la interacción entre estos dos tipos de capa. Cabe destacar que la capa Aplanar no presenta parámetros a optimizar.

3.4.4. Capa de ampliación

La capa Ampliación o, en inglés, UpSampling layer, utiliza interpolación para generar una ampliación de una imagen de entrada. Se puede utilizar dos tipos de interpolación: Vecino más cercano y bilineal. Ambos tipos de interpolación producen una ampliación de la imagen de entrada, pero realizan la estimación de los valores de los píxeles intermedios de maneras diferentes. La interpolación de vecino más cercano utiliza el valor del píxel más cercano al que se quiere estimar y la interpolación bilineal utiliza una suma ponderada de los valores de los píxeles más cercanos al que se quiere estimar.

3.4.5. Capa de agrupación

La Capa de Agrupación, o en inglés, Pooling layer se conecta con la entrada de la misma manera que una capa convolucional y posee los mismos componentes: un campo receptivo y un paso (stride), pero no posee pesos. En vez de ajustar de pesos, aplica una función directamente sobre los valores que recibe como entrada. Las funciones más comunes que se utilizan en la capa de Agrupación son la función Máximo y la función Promedio. Por ejemplo, en la Figura 3.12 se observa como el campo receptivo selecciona, utilizando la función Máximo, el valor que asigna a su salida.

La capa que se observa en la Figura 3.12 recibe el nombre de Capa de Agrupación Máxima, o en inglés, Max Pooling. Por otro lado, la Capa de Agrupación que utiliza la función promedio recibe el nombre de Capa de Agrupación Promedio o, en inglés, Average Pooling y opera de manera similar, pero calculando el promedio de los valores que se encuentran en su campo receptivo para generar su salida.

En la siguiente sección se describe el problema de utilizar algoritmos de Aprendizaje Profundo

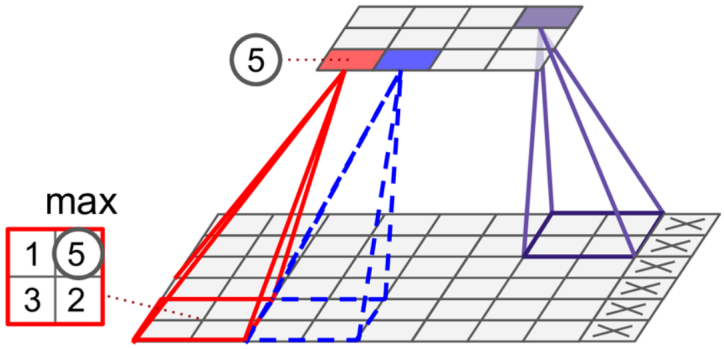

Figura 3.12: Ejemplo de Capa de Agrupación Máxima con paso dos. Imagen tomada de Aurélien (2019).

con conjuntos pequeños de datos.

3.5. Aprendizaje profundo con conjuntos pequeños de datos

En la literatura se refiere a un conjunto pequeño de datos para describir a aquel conjunto de datos que presenta una baja cantidad de ejemplos tanto para entrenar un algoritmo como para evaluar su desempeño, pero, en general, no hay un consenso sobre una cantidad exacta. En particular, para esta tesis se considera un conjunto pequeño aquel que presente menos de 500 ejemplos de entrenamiento por clase. Es decir, si se presenta un problema de clasificación de 10 categorías, cada categoría debe presentar como máximo 500 ejemplos para ser considerado como conjunto pequeño de datos.

El Aprendizaje profundo ha tenido un gran impulso en su desarrollo gracias a la disponibilidad de recursos de hardware de alta capacidad de procesamiento (D'souza et al., 2020) y la generación de grandes cantidades de datos (Chen et al., 2013; Yang et al., 2019). Sin embargo, una de sus desventajas es que requiere de grandes cantidades de datos etiquetados para alcanzar una buena exactitud (Goodfellow et al., 2016).

Esto se vuelve problemático, por un lado, debido a que el proceso de etiquetación de grandes cantidades de datos puede resultar un proceso demoroso y/o costoso (Litjens et al., 2017). Y

por otro lado, ya que hay campos que quedan fuera de la tendencia de generación masiva de datos, como el campo de la medicina (D'souza et al., 2020; Zhang et al., 2019) donde los datos son escasos, ya que están ligados a la cantidad de pacientes tratados, lo cual limita la cantidad de datos disponibles; u otros campos como la industria (Le et al., 2020; Oviedo et al., 2019) y la ciencia de materiales (Feng et al., 2019) donde existe una dificultad de recolección de datos debido a los procesos relacionados con su obtención (Barz y Denzler, 2019; Yang et al., 2019).

Estas limitaciones plantean un desafio para el aprendizaje profundo, donde se busca alcanzar los buenos resultados obtenidos usando grandes cantidades de datos en campos que presentan limitación de datos, ya sea por su costo, escasez o por su dificultad de recolección.

Dada esta problemática, en el siguiente capítulo se presentan los distintos enfoques que han enfrentado este problema.

Capítulo 4

Técnicas de aprendizaje profundo para trabajar con conjuntos pequeños de datos

4.1. Regularización

La regularización es cualquier modificación que se hace a un algoritmo de Aprendizaje Automático que intenta reducir su error de generalización pero no su error de entrenamiento (Goodfellow et al., 2016). Es decir, intenta obtener mejores resultados durante la etapa de prueba y busca reducir el sobreajuste. Particularmente, en esta tesis se utilizan dos métodos de regularización, estos son Dropout y la Normalización por Lotes, los cuales se explican a continuación.

4.1.1. Capa dropout

La primera técnica de regularización que se presenta es la Capa Dropout, propuesta por Hinton et al. (2012), cuyo objetivo es prevenir el sobreajuste de los pesos de la red neuronal forzando a las neuronas de las capas densas a depender del comportamiento grupal (Baldi y Sadowski, 2013). En la práctica, esta capa se ubica entre dos capas densas de una red neuronal profunda, desactivando una cantidad de unidades de la capa que le precede. Al desactivarse una

unidad también son desactivadas sus conexiones. Por ejemplo, en la Figura 4.1, se observa una red neuronal antes (izquierda) y después (derecha) de aplicar Dropout.

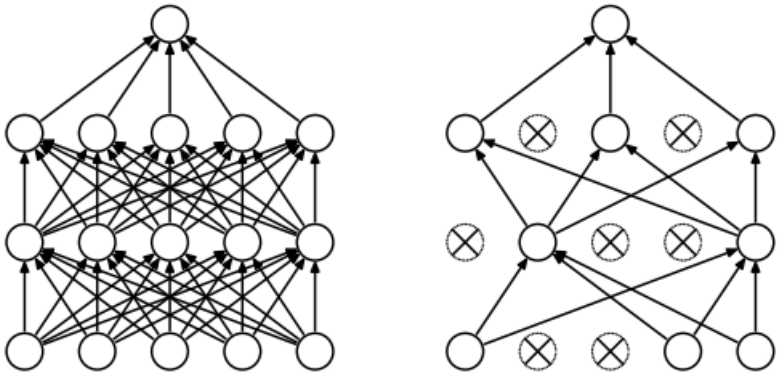

Figura 4.1: Ejemplo de Dropout en una red neuronal. Imagen tomada de Srivastava et al. (2014).

La desactivación de cada unidad solo dura una iteración del proceso de entrenamiento y va variando de manera probabilística a cada iteración de dicho proceso. En cada iteración del proceso de entrenamiento cada unidad tendrá q probabilidad de ser desactivada. Donde $q = 1 - p$ y p es la probabilidad de conservar la unidad (Baldi y Sadowski, 2013). Durante la etapa de predicción ninguna unidad es desactivada, por lo tanto, todas las unidades son utilizadas para predecir.

El uso de la capa Dropout en la red neuronal genera que cada neurona sea forzada a considerar solo las conexiones que tiene en ese determinado momento para llegar a la predicción correcta, ya que durante el entrenamiento distintas neuronas serán desactivadas, esto permite que, en general, todas las conexiones tengan la misma importancia.

4.1.2. Normalización por lote

La capa de Normalización por lote o, en inglés, Batch Normalization (BN), propuesta por Ioffe y Szegedy (2015), estandariza la salida de una capa de una red neuronal manteniendo una media de cero y una desviación estándar de uno. Esta se aplica entre las capas tanto densas como convolucionales.

En concreto, la Normalización por Lotes se aplica sobre un lote de valores de entrada $\mathcal{B} =$

$\{x_1, ..., m\}$. Primero se calcula el promedio del lote, con la Ecuación 4.1.

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \tag{4.1}$$

Luego se calcula la varianza del lote con la Ecuación 4.2.

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \tag{4.2}$$

Ya habiendo calculado el promedio y la varianza del lote, se puede normalizar, lo cual se describe en la Ecuación 4.3. Donde ϵ que se agrega para evitar el cálculo de una raíz cuadrada muy pequeña, lo cual puede provocar algunos problemas.

$$\hat{x_i} \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \tag{4.3}$$

Por último, a la salida de la capa de normalización se agregan dos parámetros γ y β que son entrenados junto con el resto de los parámetros de la red y son utilizados para desplazar o escalar la distribución previamente normalizada hacia un espacio que sea conveniente para la tarea en cuestión.

$$y_i \leftarrow \gamma\hat{x_i} + \beta \tag{4.4}$$

El uso de la capa de Normalización por Lotes actúa como un regularizador mediante la adición de una pequeña cantidad de ruido durante el proceso de entrenamiento y además permite el uso de Tasas de Aprendizaje más altas, debido a que evita que los pesos de las neuronas alcancen valores muy altos gracias a la naturaleza de la normalización (Ioffe y Szegedy, 2015).

4.2. Capa de agrupación promedio global

La capa Capa Agrupación Promedio Global o, en inglés, Global Average Pooling (GAP) realiza una operación parecida a las capas Agrupación revisadas en la sección anterior, la diferencia está en que el campo receptivo de la capa GAP es del tamaño completo de la entrada. GAP es

una operación diseñada para reemplazar las capas densas del final de una red neuronal profunda. La idea es agregar la capa GAP luego de la última capa convolucional y obtener un vector de una dimensión, con longitud igual a la cantidad de filtros que posea la última capa convolucional (Lin et al., 2013). En la Figura 4.2 se observa un ejemplo de la operación GAP.

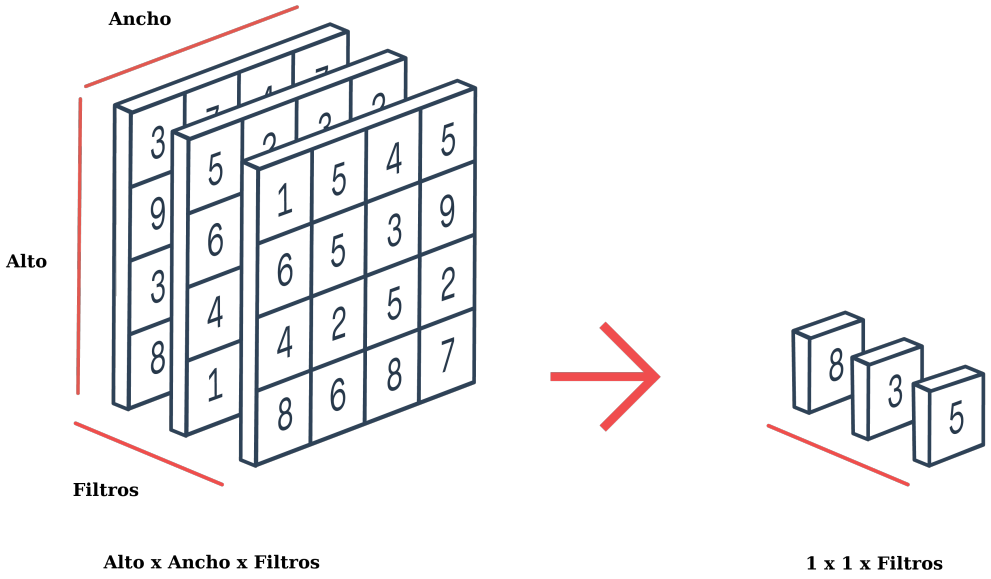

Figura 4.2: Ejemplo de operación GAP. Imagen tomada de Peltarion (2019).

La capa GAP funciona como regularizador estructural de la red convolucional ya que genera un resumen de las características de cada filtro. Además, esta capa no presenta parámetros que deban ser optimizados, por lo tanto el sobreajuste de la red no se produce en esta capa.

4.3. Convolución dilatada

La Convolución Dilatada, propuesta por Yu y Koltun (2015), es una modificación a la capa Convolucional que dilata el campo receptivo de la capa. Esta modificación permite expandir el campo receptivo sin perder resolución ni cobertura (Yu y Koltun, 2015). En la Figura 4.3 se

observa un campo receptivo de 3x3 con distintas tasas de dilatación y su deslizamiento sobre una entrada con paso (stride) uno.

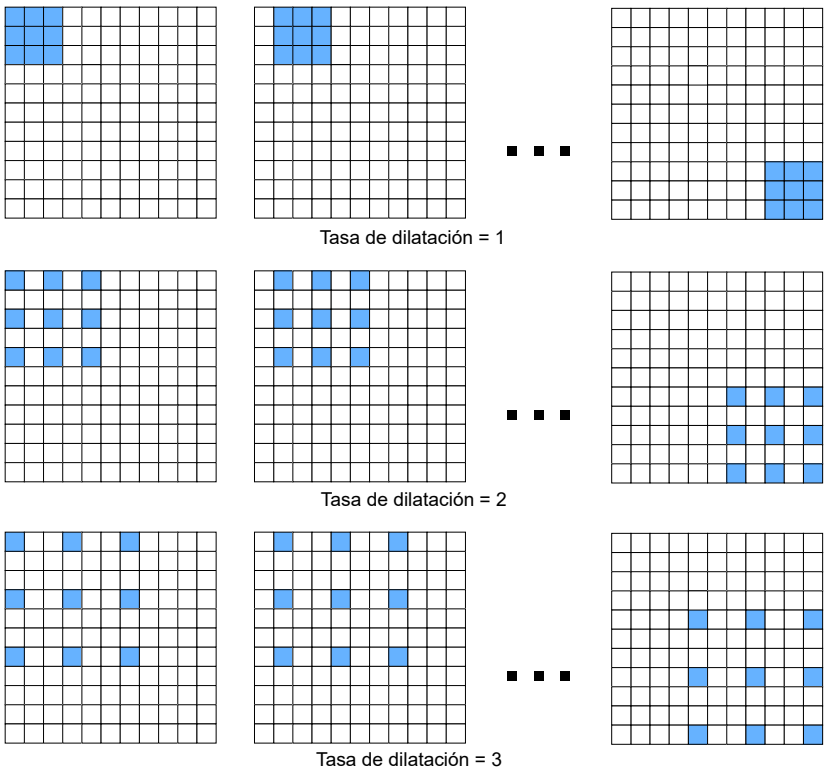

Figura 4.3: Ejemplos de campo receptivo con distintas tasas de dilatación.

El uso de la Convolución Dilatada junto con otras técnicas de regularización como Normalización por Lotes y GAP puede mejorar los resultados obtenidos en tareas de clasificación (Zhou et al., 2019).

4.4. Función de costo similitud de coseno

El uso de la función de costo Similitud de Coseno o, en inglés, Cosine Similarity, para el entrenamiento con conjuntos pequeños de datos fue propuesto en el trabajo de Barz y Denzler (2019). Esta función está basada la distancia en el ángulo entre el vector de salida del modelo que tiene las probabilidades predichas y el vector de que tiene la clasificación verdadera de la

clase (Codificación One-hot). Está definida por la siguiente ecuación:

$$\sigma_{cos}(a, b) = cos(a \angle b) = \frac{\langle a, b \rangle}{||a||_2 \cdot ||b||_2} \tag{4.5}$$

Donde a y b son vectores de K dimensiones, $\langle \cdot, \cdot \rangle$ denota el Producto Punto y $|| \cdot ||_2$ la Normalización L^2 (Barz y Denzler, 2019).

Para un vector $a = (a_1, a_2, \ldots, a_n)$ la Normalización L^2 se define como:

$$||a||_2 = \sqrt{a_1^2 + \cdots + a_n^2} \tag{4.6}$$

Para dos vectores $a = (a_1, \ldots, a_n)$ y $b = (b_1, \ldots, b_n)$ el Producto Punto se define como:

$$\langle a, b \rangle = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + \cdots + a_n b_n \tag{4.7}$$

El trabajo de Barz y Denzler (2019) concluye que el uso de Similitud de Coseno como función de costo entrenar con datos limitados obtiene mejores resultados que al utilizar la función de Entropía cruzada Categórica. Sin embargo, cuando se cuenta con un conjunto grande de datos ambas funciones de costo obtienen resultados similares. El rango de valores que entrega es [-1, 1], siendo -1 donde los vectores tienen mayor grado de similitud entre si, 1 donde se encuentra mas disimiles y 0 cuando son ortogonales.

4.5. Tasa de aprendizaje cíclica

La Tasa de Aprendizaje Cíclica (Smith, 2015) o, en inglés Cyclical Learning Rate (CLR) es una técnica de planificación de tasa de aprendizaje que asigna un valor de la tasa de aprendizaje en función del cálculo de una expresión matemática. En la Figura 4.4 se observan tres ejemplos de planificación de tasa de aprendizaje.

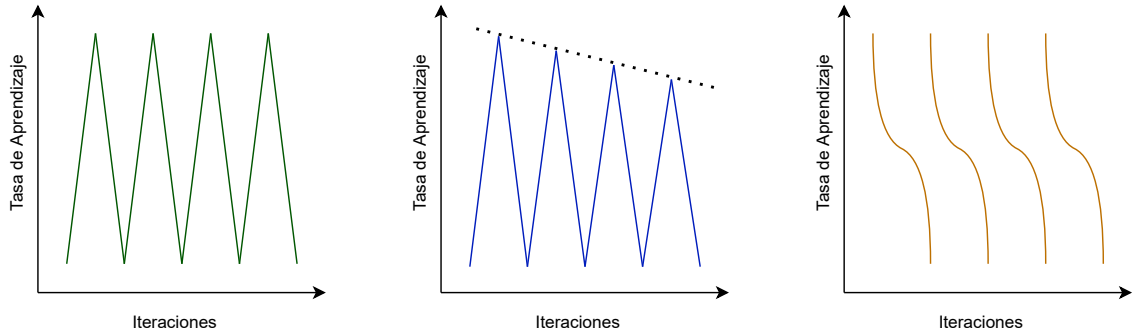

Figura 4.4: Ejemplos de planificación de tasa de aprendizaje.

La técnica de CLR comienza con una tasa de aprendizaje baja de valor lr_{min} y aumenta durante una cantidad de iteraciones s hasta llegar a un máximo de valor lr_{max}. Luego vuelve a decrecer y así sucesivamente durante la etapa de entrenamiento. CLR se define como:

$$clr(t) = lr_{min} + (lr_{max} - lr_{min}) \cdot \left(\left|1 - \frac{t}{2} - 2\left\lfloor\frac{t}{2s}\right\rfloor - 1\right|\right) \tag{4.8}$$

Donde t es la iteración del proceso de entrenamiento. En la Figura 4.5 se muestra un ejemplo de CLR.

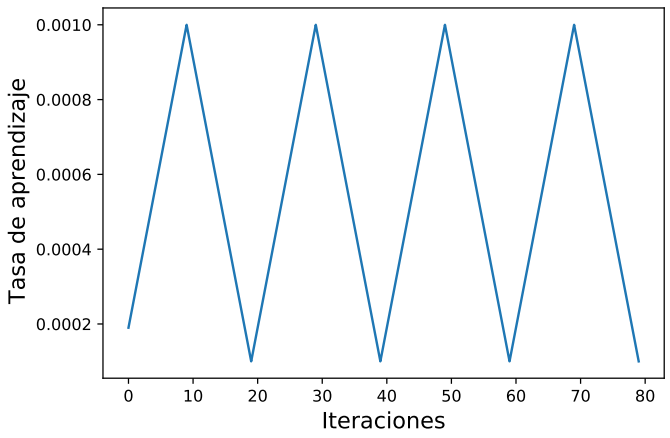

Figura 4.5: Ejemplo de CLR con los siguientes parámetros: lr_{min} de 0.0001, lr_{max} de 0.001 y s igual a 10.

En el trabajo de Barz y Denzler (2018) proponen aplicar una decadencia a CLR utilizando

la siguiente ecuación:

$$clr_{decay}(t) = \frac{clr(t)}{1 + (\delta - 1) \cdot \frac{t}{t_{max}}} \tag{4.9}$$

Donde t_{max} es la cantidad de iteraciones que durará la etapa de entrenamiento y δ es el factor de decadencia. En Figura la 4.6 se observa un gráfico del cambio de la tasa de aprendizaje durante 80 iteraciones de entrenamiento.

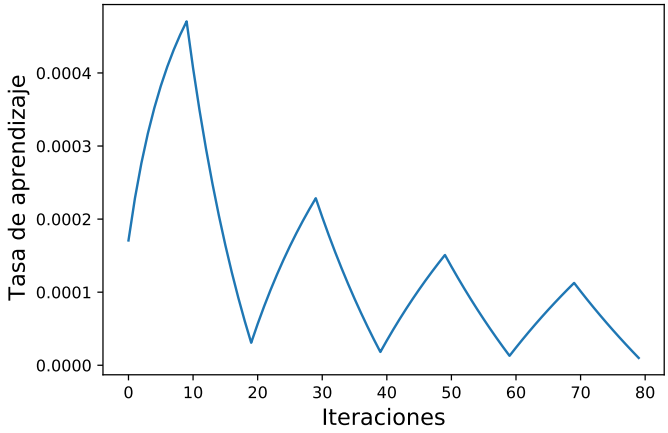

Figura 4.6: Ejemplo de decadencia sobre CLR con los mismos parámetros que la Figura 4.5, δ igual a 10 y t_{max} igual a 80.

El objetivo de modificar la tasa de aprendizaje durante el entrenamiento del algoritmo profundo es evitar que el proceso de optimización de la función de costo quede atrapado en un mínimo local.

4.6. Aumento de la cantidad datos

El Aumento de datos o, en inglés, Data Augmentation (DA) es una técnica para incrementar la cantidad de ejemplos que tiene un conjunto de datos de manera artificial. El objetivo es generar imágenes que se parezcan a las imágenes originales, idealmente no deberían ser diferenciables. Para generar las nuevas imágenes se utilizan transformaciones sobre los elementos del conjunto de datos tales como desplazamientos, rotaciones, acercamientos, alejamientos, cambios de brillo, etc. Por ejemplo, en la Figura 4.7 se puede observar un tipo de aumento sobre la imagen de un

gato, a la cual se le aplicaron rotaciones y volteo horizontal. Cabe destacar que en el ejemplo las rotaciones no han sido rellenadas, este relleno puede ser hecho de diferentes maneras. El relleno se puede hacer extendiendo el último píxel hasta el borde de la imagen, también se puede reflejar la imagen o rellenar con ceros.

Figura 4.7: Ejemplo de aumento de datos (Data Augmentation).

El Aumento de los datos se puede aplicar de tres formas: de forma estática, lo que implica aplicar las transformaciones a las imágenes antes de iniciar el proceso de entrenamiento, por lo tanto las imágenes se aumentan una sola vez; forma continua, de igual manera que la forma anterior se aplican transformaciones a cada imagen pero solo a aquellas que son seleccionadas en el Lote durante el proceso de entrenamiento, esto se realiza cada vez que se selecciona un nuevo lote de entrenamiento; la última forma es a través de la implementación de un tipo de Red Neuronal conocida como Red Neuronal Generativa Adversaria (Goodfellow et al., 2020). Estas son un tipo de Red Neuronal que se compone de de un Generador y un Discriminador. El generador se encarga de producir imágenes falsas y el discriminador se encarga de discriminar si las imágenes que recibe son falsas o reales. En el proceso de entrenamiento se busca producir un generador que produzca imágenes que el discriminador no pueda diferenciar. La última forma de Aumento de datos utiliza este tipo de red neuronal para generar imágenes similares a las que pertenecen al conjunto de datos que se desea aumentar.

El uso de esta técnica permite que el algoritmo entrenado sea más tolerante a variaciones en la posición, orientación e iluminación de los objetos de la imagen (Aurélien, 2019).

4.7. Transferencia de aprendizaje

Transferencia de Aprendizaje o, en inglés, Transfer Learning (TL) implica el uso de un modelo que fue entrenado previamente con un conjunto de datos X para una tarea X sobre un conjunto de datos Y para una tarea Y. La idea es utilizar el modelo previamente entrenado para obtener características que sean útiles sobre una tarea de dominio cercano. En la Figura 4.8 se muestra un diagrama donde se toma la base de un modelo que realiza una tarea X, lo que significa que se quita la última capa, y se agregan algunas capas al final de la base, luego se realiza en el proceso de entrenamiento para la tarea Y solo en las capas finales de la red. De esta manera se aprovechan las características que el modelo aprendió para la tarea X y se ajustan las últimas capas para la tarea Y.

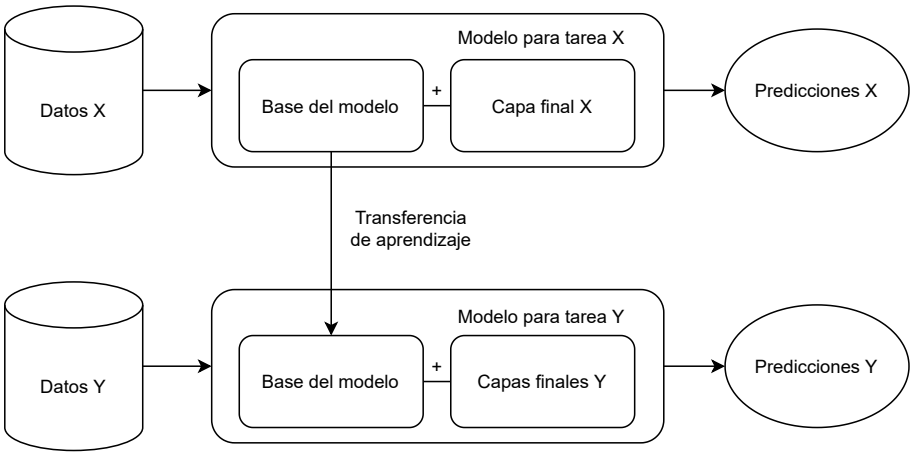

Figura 4.8: Diagrama de transferencia de aprendizaje de la tarea X a la tarea Y.

Para que la transferencia de Aprendizaje sea exitosa, los dominios tanto de las tareas X e Y como de los conjuntos de datos, deben ser cercanos.

En la Tabla 4.1 se muestran algunos de los modelos que hay disponibles en la biblioteca de código Keras. Particularmente, para esta tesis se utiliza el modelo EfficientNetB0 (Tan y Le, 2019) debido a su baja cantidad de parámetros y alto desempeño en el conjunto de datos ImageNet (Deng et al., 2009). EfficientNetB0 requiere que la imagen de entrada sea de tamaño 224 x 224 píxeles y se encuentre en formato RGB por lo que usualmente se requiere aplicar una

transformación a las imágenes del conjunto Y para que puedan ser utilizadas por el modelo X, en este caso EfficientNetB0.

Modelo	Tamaño en disco	Cantidad de Parámetros
VGG16	528 MB	138.357.544
ResNet50	98 MB	25.636.712
ResNet101	171 MB	44.707.176
ResNet152	232 MB	60.419.944
InceptionV3	92 MB	23.851.784
InceptionResNetV2	215 MB	55.873.736
DenseNet121	33 MB	8.062.504
DenseNet169	57 MB	14.307.880
DenseNet201	80 MB	20.242.984
EfficientNetB0	29 MB	5.330.571
EfficientNetB1	31 MB	7.856.239
EfficientNetB2	36 MB	9.177.569
EfficientNetB3	48 MB	12.320.535
EfficientNetB4	75 MB	19.466.823
EfficientNetB5	118 MB	30.562.527
EfficientNetB6	166 MB	43.265.143
EfficientNetB7	256 MB	66.658.687

Tabla 4.1: Modelos disponibles en la biblioteca de código Keras (Chollet y otros, 2015).

4.8. Ensamblaje de múltiples modelos

El Ensamblaje de Múltiples Modelos (EMM) es una técnica que utiliza las predicciones de varios modelos como votos, donde la clase que recibe más votos, es la clase se selecciona como la clase predicha por el EMM. En la Figura 4.9 se observa un diagrama de una predicción por EMM donde se tienen tres clases. El objetivo del Ensamblaje de Múltiples Modelos es aprovechar la sabiduría de las multitudes para realizar predicciones Kotu y Deshpande (2015).

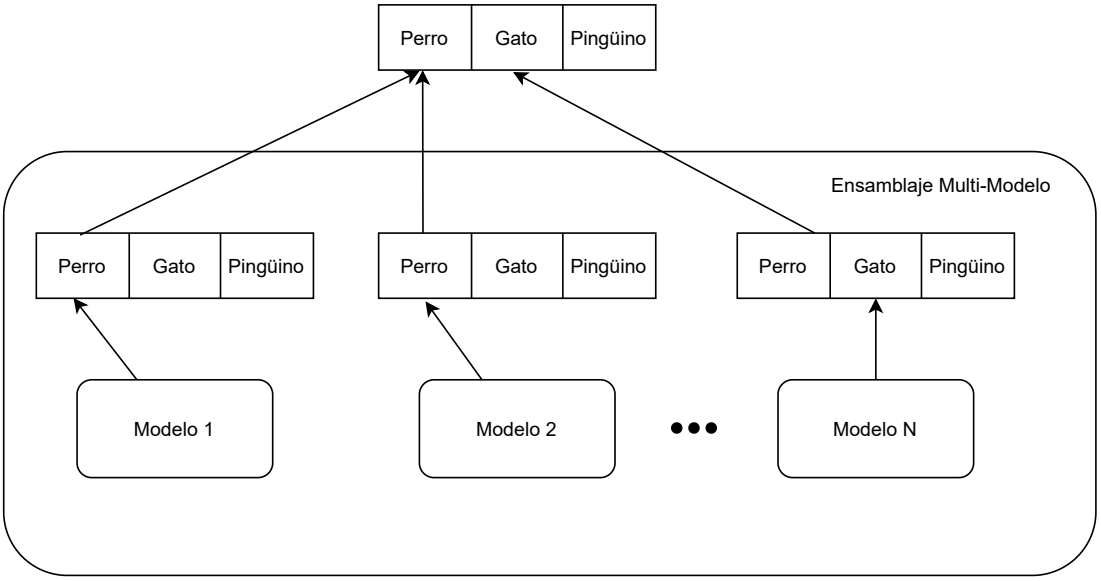

Figura 4.9: Diagrama de Ensamblaje de Múltiples Modelos.

Por ejemplo, el algoritmo de Bosque Aleatorio es un EMM, ya que está compuesto por múltiples Árboles de Decisión, donde cada uno emite un voto dada una entrada y la clase más votada es seleccionada como predicción.

Capítulo 5

Clasificación de imágenes con conjuntos pequeños de datos

Este capítulo apunta a mostrar una visión general de cómo se ha enfrentado la clasificación de imágenes con pequeños conjuntos de datos.

El trabajo de Pasupa y Sunhem (2016) compara el desempeño de algoritmos de Aprendizaje Automático Clásicos y algoritmos de Aprendizaje Profundo cuando disponen de conjuntos pequeños de datos. Los algoritmos que comparan son: Red Neuronal Prealimentada, Máquina de Soporte de Vectores y una Red Neuronal Profunda con Dropout y uso de Aumento de la cantidad de Datos. Su conjunto de datos cuenta con 500 ejemplos y 5 clases. La tarea es clasificar imágenes de caras de mujeres en 5 formas de cara diferentes. Concluyen que en general los algoritmos clásicos se desempeñan mejor que los algoritmos profundos cuando estos últimos no utilizan regularización. Sin embargo, también encuentran que al aplicar las técnicas de regularización el algoritmo profundo alcanza resultados similares a los algoritmos clásicos, por lo tanto agregan que el uso de Dropout permite mitigar el sobreajuste de la red neuronal profunda.

En el trabajo de Lu y Li (2018) se investiga el uso de Aprendizaje Profundo para clasificación de distintos tipos de barcos utilizando imágenes capturadas con radar de apertura sintética. Su conjunto de datos cuenta con 250 ejemplos con una división de 70 % para entrenamiento y 30 % para evaluación. Debido a que su conjunto de datos es pequeño utilizan las técnicas de Aumento

de la cantidad de datos, Transferencia de Aprendizaje y Dropout para mitigar el sobreajuste de su modelo. Su aumento de datos consiste en: rotar las imágenes entre 90 y 180 grados, cambiar el brillo y contraste y voltear de manera horizontal. Para aplicar la técnica de Transferencia de Aprendizaje utilizan distintos modelos previamente entrenados tales como ResNet-50, VGG-16 y DenseNet-121. Concluyen que el modelo de red neuronal profunda que proponen junto con las técnicas mencionadas obtiene mejor exactitud al compararse con otras investigaciones y que alcanza una buena capacidad de generalización.

En el trabajo de Yang et al. (2018) proponen el uso de Redes Neuronales Profundas para categorizar pares de imágenes de tratamientos dentales. Su conjunto de datos cuenta con 196 pares de imágenes del antes y después de un tratamiento dental etiquetadas por dentistas expertos en tres categorías. Para seleccionar la región de interés de cada imagen, utilizan varios métodos, entre los métodos que usan están: usar toda la imagen, selección manual y selección automática. Luego comparan utilizando Redes Neuronales Profundas, las cuales utilizan dos técnicas para mitigar su limitación en la cantidad de datos: Aumento de la cantidad de datos y Dropout. El aumento de los datos se realiza mediante la aplicación de transformaciones a las imágenes, las transformaciones fueron de tipo volteo horizontal y vertical, rotaciones y cambios en el brillo. Por último, concluyen que el método propuesto alcanza resultados comparables a los de un dentista experto. Se destaca que como métrica para medir sus resultados utilizan F1.

Por otro lado, el trabajo de Zhou et al. (2019) se estudia el efecto de utilizar Normalización por Lotes, Convolución Dilatada y capa de Agrupación de Promedio Global en algoritmos de Aprendizaje Profundo y compararlos con modelos previamente presentados como AlexNet sobre el conjunto de datos CIFAR-10. Para este trabajo se utiliza CIFAR-10 completo, que cuenta con 50.000 imágenes de entrenamiento, 10.000 imágenes de prueba y 10 clases y es definido como conjunto pequeño de datos al compararlo con el tamaño del conjunto ImageNet (alrededor de 3,2M de imágenes y 5247 clases (Deng et al., 2009)). Concluyen que el uso de las técnicas mejora ciertos aspectos del entrenamiento. En concreto, el uso de Normalización por Lotes acelera la velocidad de convergencia hacia el objetivo de la red, el uso de GAP reemplaza exitosamente las capas densas de la red reduciendo el número de parámetros de la misma y, por consiguiente, el

costo computacional del entrenamiento. Por último, el uso de la Convolución Dilatada mejora la tasa de clasificación al extender el campo receptivo de la capa convolucional.

En el trabajo de Yang et al. (2019) compara el cambio en el desempeño de Redes Neuronales Profundas al agregar una característica previamente computada a cada ejemplo de su conjunto de datos. Utilizan dos conjuntos de datos, el primero llamado *Homogenization*, el cual cuenta con 8550 imágenes de 51x51x51 y 50 clases, divididos en entrenamiento, validación y prueba con 3.819, 1.881 y 2.850 imágenes respectivamente. El segundo conjunto de datos, llamado *Crystal Plasticity*, que cuenta con 2.082 ejemplos y 24 clases, también está dividido en entrenamiento, validación y prueba con 1.262, 316 y 504 ejemplos respectivamente. La característica es calculada con una función de correlación de dos puntos, cuyo objetivo es capturar información espacial de cada imagen. Esta función es utilizada en el campo de la Informática de Materiales, para más información sobre ella y sus funciones se recomienda revisar el trabajo en cuestión. Concluyen que al agregar estas características logran guiar el proceso de entrenamiento, lo cual permite mejorar el desempeño del modelo y además ayuda a que las predicciones del modelo sean más explicables.

En la misma línea del trabajo anterior, Zhang et al. (2019) añade características a cada imagen para aumentar la información que los algoritmos pueden usar para clasificar. En particular, utilizan algoritmos de Aprendizaje Profundo para diferenciar lesiones malignas de benignas en tomografías computarizadas. Utilizan dos conjuntos de datos, *Colorectal Polyps* y *Lung Nodules* con 63 y 67 ejemplos respectivamente. Una de las características que añaden a la imagen es el Patrón Local Binario (LBP) de cada una, que es una función que se utiliza para extraer información sobre las texturas que se presentan en una determinada imagen. Otra de las características es llamada Características de Gradiente y se calcula utilizando el algoritmo de Canny que se utiliza para detectar bordes en las imágenes. Por último, la última característica que añaden son los Histogramas de Gradientes Orientados que es otro descriptor de características utilizado en imágenes. Para mayor información sobre las funciones que utilizan para calcular las características de cada imagen se sugiere al lector la revisión del trabajo de Zhang et al. (2019). Concluyen que dado que sus conjuntos de datos son pequeños, añadir directamente características al proceso

de entrenamiento mejoró el desempeño de sus algoritmos comparado con redes neuronales a las que no se agregó dichas características. Además determinan que LBP resulta ser más importante que las otras dos características que añaden.

Con técnicas similares, en el trabajo de Lee et al. (2019) desarrollan un algoritmo de Aprendizaje Profundo para detectar hemorragia intracraneal aguda (ICH), sangrado dentro del cráneo, en tomografías computarizadas de cabeza de distintos pacientes. Su conjunto de datos cuenta de 904 casos y seis categorías (cinco tipos de hemorragias y el caso en que no se presenta ninguna), los datos se dividen en 704 para el entrenamiento y 200 para la etapa de prueba. Adicionalmente, recopilan alrededor de 200 casos para un segundo conjunto de prueba. Dado que la cantidad de datos que tienen es limitada, utilizan las siguientes técnicas para mitigar el problema: Aumento de la cantidad de Datos utilizando transformaciones sobre las imágenes tales como rotaciones, acercamientos y desplazamientos. También, implementan la técnica de Transferencia de Aprendizaje mediante el uso de los modelos, previamente entrenados con el conjunto ImageNet, VGG-16, ResNet-50, Inception-v3 e Inception-ResNet-v2. Además, añaden conocimiento del dominio mediante el cálculo de dos características relativas al tipo de dato que usan (tipo médico), estas son: *Multi-window conversion* y *Slice Interpolation*. Por último, utilizan un Ensamblaje de Múltiples Modelos para utilizar los modelos de todos los experimentos que realizaron. Realizan cuatro experimentos, en cada experimento van agregando más datos al proceso de entrenamiento. De los experimentos que realizan, todos utilizan Transferencia de Aprendizaje como base y de esos, los experimentos que utilizan todas las técnicas propuestas son los que mayor desempeño alcanzan. Sin embargo, estos son superados por el Ensamblaje de los todos los modelos que entrenaron. Como métricas para evaluar el desempeño de sus experimentos utilizan precisión promedio. Por último, concluyen que al agregar más información al proceso de entrenamiento, el desempeño de los algoritmos aumenta y comparan el aumento en el desempeño al aplicar el ensamblaje de múltiples modelos a contar con varias personas haciendo clasificaciones manuales.

En el trabajo de Gozes y Greenspan (2019) utilizan Transferencia de Aprendizaje y Aumento de la cantidad de Datos para desarrollar un algoritmo de Aprendizaje Profundo que detecte tuberculosis en imágenes de rayos X. Cuentan con tres conjuntos de datos, ChestXRay14, Shenzen

TB y Montgomery TB. El primer conjunto cuenta con 112.000 imágenes, las cuales dividen en 104.266 para entrenamiento, 6.336 para validación y 1.518 para prueba. El segundo, cuenta con 462 imágenes de entrenamiento, 100 de validación y 100 de prueba. Y el tercero cuenta con 138 imágenes más que utilizan solo para validación. Primero, entrenan un modelo pre-entrenado (DenseNet-121) con el conjunto de datos ChestXRay14 y lo utilizan para predecir 14 patologías. Además, añaden los metadatos de las imágenes para predecir la posición y edad del paciente. Segundo, toman el modelo anterior y vuelven a aplicar Transferencia de Aprendizaje sobre el conjunto de datos Shenzen TB, con el objetivo de predecir Tuberculosis, en esta etapa, como el conjunto Shenzen TB es pequeño, utilizan un Aumento de la cantidad de Datos solo con volteos horizontales. Por último, utilizan el tercer conjunto de datos (Montgomery TB) para validar los resultados. Sus resultados muestran que haber entrenado el modelo DenseNet-121 con ChestXRay14 y utilizar el modelo resultante para clasificar sobre el conjunto Shenzen les entrega el mejor desempeño obtenido sobre ese conjunto. Sin embargo, al sumar el tercer conjunto de prueba a la etapa de evaluación, los resultados cambian y el modelo que mejor desempeño obtiene es otro modelo propuesto en otro trabajo. Concluyen que el entrenamiento con ChestXRay14 facilita una mejor Transferencia de Aprendizaje para conjuntos pequeños de datos relativos a tuberculosis.

Barz y Denzler (2019) comparan la diferencia de desempeño de los algoritmos de Aprendizaje Profundo al utilizar las funciones de costo Entropía cruzada Categórica y Similitud de Coseno. Utilizan seis conjuntos pequeños de datos de imágenes: CUB, NAB, Stanford Cars, Oxford Flowers, MIT Indoor Scenes y CIFAR-100, cuya descripción se puede observar en la Tabla 5.1. Y también, utilizan un conjunto de datos de texto llamado AG News, que cuenta con 127.600 artículos de noticias y cuatro categorías, se divide en 120.000 artículos para entrenamiento y 7.600 para validación. Proponen como técnica para enfrentar la limitación en la cantidad de datos el uso de la función de costo Similitud de Coseno, la cual se utiliza para calcular la pérdida durante el entrenamiento del algoritmo. También, utilizan la Decadencia Cíclica de la Tasa de Aprendizaje para modificar el valor de la Tasa de Aprendizaje durante el entrenamiento. La arquitectura del algoritmo que entrenan es ResNet-110 para el conjunto CIFAR-100 y ResNet-50 para todos

	Total	Entrenamiento	Prueba	Clases
CUB	11.788	5.994	5.794	200
NAB	48.562	23.929	24.633	555
Stanford Cars	16.185	8.144	8,041	196
Oxford Flowers 102	8.189	2.040	6,149	102
MIT Indoors Scenes	6.700	5,360	1,340	67
CIFAR-100	60.000	50.000	10.000	100

Tabla 5.1: Detalles de los conjuntos de datos utilizados en el trabajo de Barz y Denzler (2019).

los otros conjuntos. Concluyen que usar Similitud de Coseno para entrenar con datos limitados obtiene mejores resultados que al utilizar la función Entropía cruzada Categórica. Sin embargo, cuando se cuenta con un conjunto grande de datos ambas obtienen resultados similares.

Brigato y Iocchi (2020) estudian el impacto de la complejidad de un algoritmo de Aprendizaje Profundo sobre el desempeño obtenido cuando se cuenta con pocos ejemplos por clase. Utilizan tres conjuntos de datos: Fashion MNIST, SVHN y CIFAR-10. Para cada uno de los conjuntos de datos seleccionan ocho subconjuntos de 10, 20, 30, 40, 80, 260, 320, 640 y 1.280 ejemplos por clase. Comparan tres métodos diferentes, uno propuesto por ellos basado en autoenconders, un tipo de algoritmo de aprendizaje automático no supervisado; otro basado en redes neuronales convolucionales y un último método basado en la arquitectura ResNet-20. Utilizan el aumento de la cantidad de Datos con transformaciones de tipo volteo horizontal, rotaciones entre 0^{o} y 20^{o} y desplazamientos horizontales y verticales, también utilizan Dropout. Sus resultados muestran que los modelos de menor complejidad son menos propensos a presentar sobreajuste y que Dropout actúa como un buen regularizador. Sin embargo, plantean que el Aumento de los Datos no siempre es beneficioso y que es altamente dependiente de la forma en que se aplica, recomendando aplicarlo de forma estática.

En el trabajo de Le et al. (2020) proponen un marco para enfrentar la limitación en la cantidad de datos disponibles para el entrenamiento de Algoritmo de Aprendizaje Profundo basado en Aumento de los datos, Transferencia de Aprendizaje y Ensamblaje de Múltiples Modelos. Se utilizan dos conjuntos de datos, el primero se compone de 8.595 imágenes tipo RGB de hojas decorativas de una línea de producción y cuenta con seis categorías. El segundo conjunto de datos

se compone de 7.606 imágenes de rayos X en formato RGB de defectos en soldaduras con cuatro categorías. Ambos se dividen en 80 % para la etapa de entrenamiento y 20 % para la etapa de prueba. La cantidad de datos de cada conjunto ha sido aumentada utilizando dos técnicas, la primera basada en la generación de defectos aleatorios en imágenes y en transformaciones de la imagen, es decir, rotaciones, volteos, desplazamientos, etc. La segunda técnica que utilizan para aumentar los datos requiere el entrenamiento de un tipo de red neuronal conocida como Redes Generativas Adversarias, compuesta de un generador y un discriminador, donde el generador es entrenado para engañar al discriminador y el discriminador entrenado para discriminar las imágenes producidas por el generador, indicando si la imagen es real o generada. Utilizan Redes Generativas Adversarias para generar nuevos ejemplos para sus conjuntos de datos. Luego, para el proceso de entrenamiento, utilizan Transferencia de Aprendizaje, basándose en los modelos Inception y MobileNet. Por último, desarrollan un Ensamblaje con los modelos entrenados. Concluyen que el uso de las técnicas propuestas reduce la cantidad de falsos positivos al momento de predecir y que el marco propuesto fue exitoso para la detección de defectos en hojas decorativas y defectos en soldaduras.

De igual manera, en el trabajo de Rajpurkar et al. (2020) se propone una arquitectura de Red Neuronal Convolucional que llaman ApendiXNet para clasificar tomografías computarizadas de abdomen/pelvis y detectar apendicitis. Utilizan un conjunto de datos compuesto 646 de tomografías de abdomen/pelvis de las cuales 438 son utilizadas para el entrenamiento, 106 para la etapa de prueba, 102 para validación. Para mitigar su limitación en la cantidad de datos utilizan dos técnicas. La primera técnica que aplican es Normalización por Lotes en su arquitectura de Red Neuronal. La segunda técnica que aplican es Transferencia de Aprendizaje, donde pre-entrenan su arquitectura con el conjunto de datos Kinetics[1], que cuenta con aproximadamente 500.000 vídeos de 10 segundos de largo y 600 clases. Luego, sobre este modelo entrenan la tarea de detección de apendicitis con su conjunto pequeño de datos. Sus resultados muestran que al utilizar la Transferencia de Aprendizaje su modelo aumenta su desempeño desde 56,9 % de exactitud a 72,5 %. Lo anterior les lleva a concluir que la aplicación de Transferencia de Aprendizaje permite

[1] https://deepmind.com/research/open-source/kinetics

compensar por la limitación en su conjunto de datos.

D'souza et al. (2020) estudian la influencia de la estructura de la red neuronal sobre el desempeño de esta y si es que la estructura óptima es determinada por el tamaño del conjunto de datos o por la naturaleza de los datos. Para sus experimentos utilizan tres conjuntos de datos, estos son: MNIST, CIFAR-10 y Mitosis. MNIST contiene 70.000 imágenes de dígitos escritos a mano con una división de 60.000 para entrenamiento, 10.000 imágenes para prueba y 10 clases. Por otro lado, CIFAR-10 cuenta con 60.000 imágenes de distintos objetos y animales con una división de 50.000 imágenes para entrenamiento, 10.000 imágenes para prueba y 10 clases. El último conjunto de datos que usan, Mitosis, cuenta con 4.290 imágenes en formato TIFF y dos clases. Estos conjuntos de datos los dividieron en subconjuntos, MNIST en tres subconjuntos, CIFAR-10 y Mitosis en un subconjunto, estas divisiones se pueden observar en la Tabla 5.2.

<table>
<tr><td>Conjunto de datos</td><td colspan="6">subconjuntos</td></tr>
<tr><td rowspan="3">MNIST</td><td colspan="2">1.000</td><td colspan="2">500</td><td colspan="2">100</td></tr>
<tr><td>Entrenamiento</td><td>Prueba</td><td>Entrenamiento</td><td>Prueba</td><td>Entrenamiento</td><td>Prueba</td></tr>
<tr><td>800</td><td>200</td><td>400</td><td>100</td><td>60</td><td>40</td></tr>
<tr><td rowspan="3">CIFAR-10</td><td colspan="6">5.000</td></tr>
<tr><td colspan="4">Entrenamiento</td><td colspan="2">Prueba</td></tr>
<tr><td colspan="4">4.000</td><td colspan="2">1.000</td></tr>
<tr><td rowspan="3">Mitosis</td><td colspan="6">4.290</td></tr>
<tr><td colspan="2">Entrenamiento</td><td colspan="2">Validación</td><td colspan="2">Prueba</td></tr>
<tr><td colspan="2">2.500</td><td colspan="2">800</td><td colspan="2">990</td></tr>
</table>

Tabla 5.2: Distribución de subconjuntos que utiliza D'souza et al. (2020).

Para encontrar la combinación estructural óptima para cada conjunto, probaron 7.103 estructuras diferentes para MNIST, 2.821 estructuras para CIFAR-10 y 2.599 para el conjunto Mitosis. Luego, para encontrar la combinación óptima de parámetros de dicha estructura realizan una búsqueda iterativa por los valores de cada parámetro en potencias de dos (16, 32, 64, ...). Por último presentan las estructuras que menor error obtuvieron para cada subconjunto. Concluyen

que, como las estructuras para cada conjunto son, en sus palabras, bastante diferentes, la estructura óptima está dirigida por el tipo de dato que intenta clasificar. Por lo anterior, mencionan que la Transferencia de Aprendizaje que se aplica a una tarea podría no funcionar correctamente para otra tarea, cuando los tipos de datos son muy diferentes.

En la Tabla 5.3 se observa un resumen de los trabajos presentados en este capítulo, donde se puede observar la cantidad de datos con los que trabajan y qué técnicas aplican para enfrentar el problema de tener con conjunto pequeño de datos para entrenar un Algoritmo de Aprendizaje Automático.

Trabajo	Tamaño conjunto de datos	Técnicas
Pasupa y Sunhem (2016)	500	Aumento de los Datos, Dropout
Lu y Li (2018)	250	Transferencia de Aprendizaje, Aumento de los Datos, Dropout
Yang et al. (2018)	196	Aumento de los Datos, Dropout
Yang et al. (2019)	8550, 2082	Añadir conocimiento del dominio
Zhang et al. (2019)	63, 67	Añadir conocimiento del dominio
Zhou et al. (2019)	60.000	Normalización por Lotes, Convolución Dilatada, Agrupación de Promedio Global
Lee et al. (2019)	904, 138	Transferencia de Aprendizaje, Añadir conocimiento del dominio, Ensamblaje de Modelos
Gozes y Greenspan (2019)	112.000, 662, 138	Transferencia de Aprendizaje, Aumento de los Datos
Barz y Denzler (2019)	11.788, 48.562, 16.185, 8.189, 6.700, 60.000, 127.600	Cambian la función de costo, Decadencia cíclica de Tasa de Aprendizaje
Brigato y Iocchi (2020)	10, 20, 30, 40, 80, 160, 320, 640, 1280	Aumento de los Datos, Dropout
Le et al. (2020)	8595, 7606	Transferencia de Aprendizaje, Aumento de los Datos, Ensamblaje de Modelos
Rajpurkar et al. (2020)	646	Transferencia de Aprendizaje, Normalización por Lotes
D'souza et al. (2020)	1.000, 500, 100, 5.000, 4.290	Búsqueda de estructura de red neuronal óptima

Tabla 5.3: Resumen de las técnicas que cada trabajo aplica para enfrentar el problema de contar con un conjunto pequeño de datos.

En base a esto en el siguiente Capítulo se describen los experimentos a desarrollar para probar las distintas técnicas propuestas en la literatura y sus respectivos resultados.

Capítulo 6

Experimentación y resultados

Este capítulo presenta los experimentos a realizar y los resultados obtenidos. Se describen los conjuntos de datos utilizados, el preprocesamiento aplicado a los datos, los algoritmos a comparar, las técnicas aplicadas, las métricas empleadas y la configuración del sistema utilizado para el entrenamiento y prueba. Respecto de los resultados, se presentan de dos maneras: primero, se presentan los parámetros utilizados, la arquitectura de la red y los resultados en las métricas de cada algoritmo por sí solo; luego, se presentan los resultados agregados por métrica, facilitando la comparación entre algoritmos.

6.1. Experimentación

6.1.1. Conjuntos de datos

Los experimentos se realizan sobre tres conjuntos de datos:

- MNIST[1]
- Fashion MNIST [2]
- CIFAR-10 [3]

[1] http://yann.lecun.com/exdb/mnist/
[2] https://github.com/zalandoresearch/fashion-mnist
[3] https://www.cs.toronto.edu/~kriz/cifar.html

(a) MNIST (b) Fashion MNIST (c) CIFAR-10

Figura 6.1: Ejemplos de los conjuntos de datos.

6.1.1.1. MNIST

El conjunto de datos MNIST cuenta con 70.000 imágenes, de las cuales 60.000 son destinadas para el entrenamiento y 10.000 son destinadas para la etapa de evaluación. Cada imagen representa un dígito del sistema numérico de base 10, por consiguiente este conjunto presenta diez clases, una para cada dígito. Las imágenes tienen un tamaño de 28 x 28 píxeles y se encuentran en escala de grises, por lo que solo tienen un canal de información. En la Figura 6.1(a) se muestran tres imágenes de este conjunto junto con su respectiva etiqueta.

6.1.1.2. Fashion MNIST

El conjunto de datos Fashion MNIST consiste, al igual que el conjunto MNIST, de 60.000 imágenes de entrenamiento, 10.000 para la etapa evaluación y 10 clases. Las imágenes hacen referencia a diferentes tipos de vestuario y las clases son las siguientes: T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag y Ankle boot. Las imágenes son de tamaño 28 x 28 píxeles y se encuentran en escala de grises. En la Figura 6.1(b) se pueden observar tres ejemplos del conjunto de datos y sus etiquetas.

6.1.1.3. CIFAR-10

El conjunto de datos CIFAR-10 cuenta con 50.000 imágenes para entrenamiento, 10.000 imágenes para prueba y 10 clases. Las imágenes son de distintos vehículos y animales y sus etiquetas son: airplane, automobile, bird, cat, deer, dog, frog, horse, ship y truck. Cada imagen tiene un tamaño de 32 x 32 píxeles en formato RBG, es decir, cuenta con tres canales de información, uno para el color rojo (R), otro para el color azul (B) y un último canal para el color verde (G). En la Figura 6.1(c) se presentan tres ejemplos de las imágenes que componen este conjunto de datos, cada una junto a su respectiva etiqueta.

6.1.1.4. Subconjuntos de datos y preprocesamiento

Dado que se quiere evaluar el desempeño de distintos algoritmos y técnicas cuando se cuenta con un conjunto pequeño de datos, se crean cuatro subconjuntos para cada conjunto de datos. Los subconjuntos tienen los siguientes tamaños: 100 (10 imágenes por clase), 500 (50 imágenes por clase), 2500 (250 imágenes por clase), 5000 (500 imágenes por clase). Cada subconjunto es seleccionado al azar desde el conjunto dispuesto para el entrenamiento de cada conjunto de datos. Estos subconjuntos serán utilizados para el entrenamiento de los algoritmos. Por otro lado, para la etapa de prueba se utilizará el conjunto de evaluación que cada conjunto de datos tiene. Es decir, se entrenará con un subconjunto de tamaño pequeño, pero se evaluará en el conjunto de pruebas completo.

Luego de la selección de los subconjuntos se preprocesan los datos. El preprocesamiento consiste en la normalización del valor de cada píxel de las imágenes. Se utiliza la normalización mínimo-máximo utilizando la Ecuación 6.1.

$$X = \frac{x_i - min(X)}{max(X) - min(X)} \tag{6.1}$$

Donde X es un conjunto de datos y x_i es un elemento de dicho conjunto.

Para el caso de las imágenes, la aplicación de esta normalización significa cambiar el rango de los valores de cada píxel de [0, 255] a [0, 1]. También, como parte del preprocesamiento se

aplica la codificación One-hot a las etiquetas de cada conjunto de datos.

6.1.2. Algoritmos y técnicas

Las técnicas a implementar son: Dropout, Agrupación de Promedio Global, Normalización por lotes, Similitud de Coseno, Convolución Dilatada, Decadencia Cíclica de Tasa de Aprendizaje, Transferencia de aprendizaje, Aumento de datos y Ensamblaje de Múltiples Modelos.

Se implementan 13 algoritmos para cada conjunto de datos y se entrenan con los cuatro subconjuntos para notar el cambio en desempeño al aumentar la cantidad de datos disponibles para entrenamiento. Los algoritmos implementados son:

- Algoritmos clásicos
  - (1) Árbol de decisión
  - (2) Bosque aleatorio
  - (3) Máquina de soporte de vectores
- Algoritmos profundos
  - (4) Red neuronal profunda base
  - (5) Red neuronal profunda con Dropout
  - (6) Red neuronal profunda con Normalización por lotes
  - (7) Red neuronal profunda con Convolución dilatada
  - (8) Red neuronal profunda con Similitud de coseno
  - (9) Red neuronal profunda con Agrupación de promedio global
  - (10) Red neuronal profunda con Tasa de aprendizaje cíclica
  - (11) C1: Combinación de técnicas (5 al 10)
  - (12) C2: Añade Transferencia de aprendizaje y Aumento de datos a la combinación de técnicas anterior.
  - (13) Ensamblaje de redes neuronales (4 al 12)

La arquitectura de las Redes Neuronales Profundas de base para los conjuntos MNIST y CIFAR-10 están basadas en las arquitecturas presentadas en el trabajo de D'souza et al. (2020), ya que en su trabajo utilizan estos conjuntos de datos con subconjuntos de tamaño similar. La

arquitectura de la Red Neuronal Profunda de base utilizada para Fashion MNIST se define a través de prueba y error.

La definición de parámetros de los algoritmos clásicos se realiza utilizando Búsqueda en Cuadrícula[4], que es un método que prueba un conjunto determinado de parámetros e indica los resultados para cada combinación. Por otro lado, para los algoritmos de aprendizaje profundo, los parámetros se ajustan mediante prueba y error durante varias iteraciones del proceso de entrenamiento.

A continuación se indica en qué parte de cada algoritmo se aplica cada técnica de aprendizaje profundo:

- Para la red neuronal con Dropout, se añade la capa Dropout entre las capas densas de la red neuronal base.
- Para la red neuronal con Agrupación de Promedio Global, se cambia la capa Aplanar por la capa de Agrupación de Promedio Global y se eliminan las capas densas, solo se deja la última capa densa que realiza la predicción.
- Para la red neuronal con Normalización por lotes, se añade la capa de Normalización por lotes en las capas más cercanas a la entrada de la red neuronal.
- Para la red neuronal con Similitud de Coseno, se cambia la función de costo Entropía cruzada categórica por la función de costo Similitud de Coseno.
- Para la red neuronal con Convolución Dilatada, se modifican las capas convolucionales de la red neuronal base para que utilicen la dilatación en sus respectivos campos receptivos.
- Para la red neuronal con Decadencia Cíclica de Tasa de Aprendizaje, se añade la función de decadencia cíclica para que durante el entrenamiento se vaya modificando la tasa de aprendizaje.
- Para la red neuronal con Combinación de técnicas (C1), se añade Agrupación de Promedio Global en reemplazo de las capas densa (se reemplazan todas menos la última, que es la encargada de entregar la salida de predicción), se añade Dropout entre la capa de Agrupación de Promedio Global y la última capa densa de predicción, se añade la capa

[4] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

de Normalización por lotes en las capas más cercanas a la entrada de la red neuronal, se cambia la función de costo Entropía cruzada categórica por la función de costo Similitud de Coseno, se añade dilatación a las capas convolucionales y se aplica la función de decadencia cíclica para que durante el entrenamiento.

- Para la red neuronal con Combinación de técnicas más Transferencia de aprendizaje y Aumento de datos (C2), se reemplazan las capas convolucionales el modelo EfficientNetB0, por lo tanto esta red neuronal no presenta convolución dilatada. También, para estos experimentos con transferencia de aprendizaje, en los subconjuntos de MNIST y Fashion MNIST, se debe añadir dos canales de información, ya que el modelo transferido recibe como entrada una imagen de tres canales. Esto se realiza copiando el canal en escala de grises dos veces para completar los tres canales requeridos. Para los datos del conjunto CIFAR-10 no es necesario realizar esto debido a que son de tipo RGB, por lo que ya presentan tres canales de información. Además, para estos experimentos también se utiliza la capa Ampliar, ya que la entrada que recibe la red es de tamaño 224 x 224 píxeles, por lo que se requiere ampliar cada imagen antes de que pueda ser utilizada como entrada. Para el entrenamiento de este algoritmo, se entrenan las últimas seis capas del modelo EfficientNetB0 junto con las capas agregadas para adaptación de la tarea en cuestión. Respecto del aumento de datos, se aplica de forma continua, durante el entrenamiento del algoritmo, realizando rotaciones, desplazamientos y volteo horizontal.
- Por último, para el Ensamblaje de Múltiples Modelos se utilizan todos los algoritmos de redes neuronales profundas para realizar predicciones y se obtiene una respuesta definitiva basada en la clase que obtiene más votos.

Cada algoritmo es entrenado con los cuatro subconjuntos de cada conjunto de datos (MNIST, Fashion MNIST y CIFAR-10) para notar el cambio en el desempeño al aumentar los datos disponibles. Se utiliza la misma arquitectura de los algoritmos para los distintos subconjuntos de un conjunto de datos, pero diferentes valores de parámetros al cambiar de subconjunto de datos. Los valores de los parámetros de cada experimento junto con la arquitectura de la red neuronal se indican en una tabla en la sección de resultados.

6.1.3. Métricas a utilizar para la evaluación

Cada experimento se evalúa con las siguientes métricas:

- Exactitud
- Precisión
- Recall
- F1
- Coeficiente de Correlación de Matthews (MCC)

Respecto de los tiempos de ejecución de los algoritmos, estos no se consideran como una métrica comparativa para esta tesis debido a que el entorno de ejecución de los algoritmos era compartido con otros usuarios, por lo que existe discrepancia entre la carga que presenta el sistema al momento de ejecutar cada uno de los experimentos. Sin embargo, si es un factor a considerar al momento de decidir cuántos experimentos realizar.

A modo de resumen, se presenta la Figura 6.2 en la cual se observa una representación visual de los componentes de la experimentación.

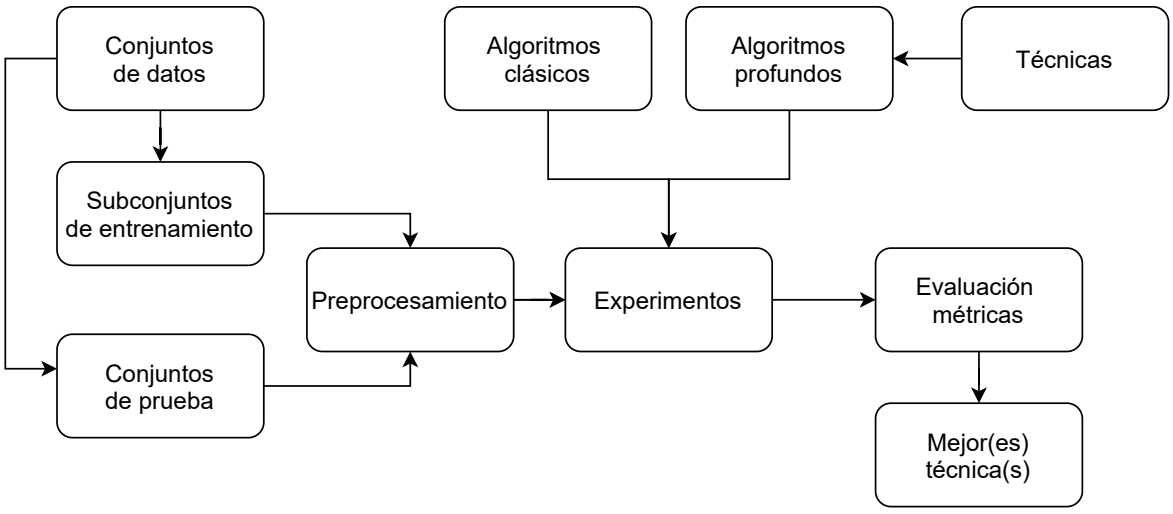

Figura 6.2: Diagrama de la experimentación realizada.

6.1.4. Configuración del sistema empleado

Para la ejecución de los experimentos se utiliza el servidor de Postgrado de la Universidad del Bío-Bío cuyas características son las siguientes:

- Sistema operativo Ubuntu 18.04.03 (4.15.0.101-generic kernel)
- 2 procesadores Intel Xeon Gold 5118 @ 2.30GHz
- 62,6 GB de Memoria RAM

Por último, el código de los experimentos es implementado en lenguaje de programación Python 3.6.9 y se utilizan las siguientes bibliotecas: Scikit-learn (Pedregosa et al., 2011), Tensorflow (Abadi et al., 2016), Keras (Chollet y otros, 2015), entre otras. El código implementado se encuentra disponible en Github[5].

En la siguiente sección se presentan los resultados de los experimentos descritos en esta sección.

6.2. Resultados

En esta sección se presentan los resultados de los experimentos para cada subconjunto de datos agregados por métrica, además, se grafican estos resultados para contrastar el desempeño de cada algoritmo. La configuración de cada algoritmo, tal como: arquitectura de la red, parámetros de configuración y técnicas utilizadas se puede encontrar en el Apéndice A.

6.2.1. Subconjuntos de MNIST

Exactitud Los resultados agregados por exactitud se presentan en la Tabla 6.1, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.3 se puede observar un gráfico de barras de la exactitud obtenida por cada algoritmo y técnica.

[5] https://github.com/Gonm1/smalldata

Exactitud				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,418 (-38,53 %)	0,601 (-34,67 %)	0,74 (-23,55 %)	0,789 (-19,65 %)
Bosque aleatorio	0,749 (10,15 %)	0,89 (-3,26 %)	0,933 (-3,62 %)	0,946 (-3,67 %)
Máquina de soporte de vectores	0,792 (16,47 %)	0,903 (-1,85 %)	0,946 (-2,27 %)	0,96 (-2,24 %)
Red Neuronal Profunda	0,68 (0,0 %)	0,92 (0,0 %)	0,968 (0,0 %)	0,982 (0,0 %)
Dropout	0,746 (9,71 %)	0,903 (-1,85 %)	0,944 (-2,48 %)	0,979 (-0,31 %)
Agrupación Promedio Global	0,803 (18,09 %)	0,902 (-1,96 %)	0,982 (1,45 %)	0,98 (-0,2 %)
Normalización por lotes	0,674 (-0,88 %)	0,914 (-0,65 %)	0,94 (-2,89 %)	0,954 (-2,85 %)
Similitud de coseno	0,703 (3,38 %)	0,894 (-2,83 %)	0,961 (-0,72 %)	0,961 (-2,14 %)
Convolución dilatada	0,693 (1,91 %)	0,883 (-4,02 %)	0,957 (-1,14 %)	0,972 (-1,02 %)
Decadencia cíclica de tasa de aprendizaje	0,692 (1,76 %)	0,89 (-3,26 %)	0,967 (-0,1 %)	0,978 (-0,41 %)
Combinación de técnicas (C1)	0,804 (18,24 %)	0,938 (1,96 %)	0,973 (0,52 %)	0,976 (-0,61 %)
Combinación de técnicas (C2)	0,796 (17,06 %)	0,93 (1,09 %)	0,967 (-0,1 %)	0,981 (-0,1 %)
Ensamblaje	0,868 (27,65 %)	0,96 (4,35 %)	0,986 (1,86 %)	0,99 (0,81 %)

Tabla 6.1: Comparación de resultados de exactitud sobre MNIST.

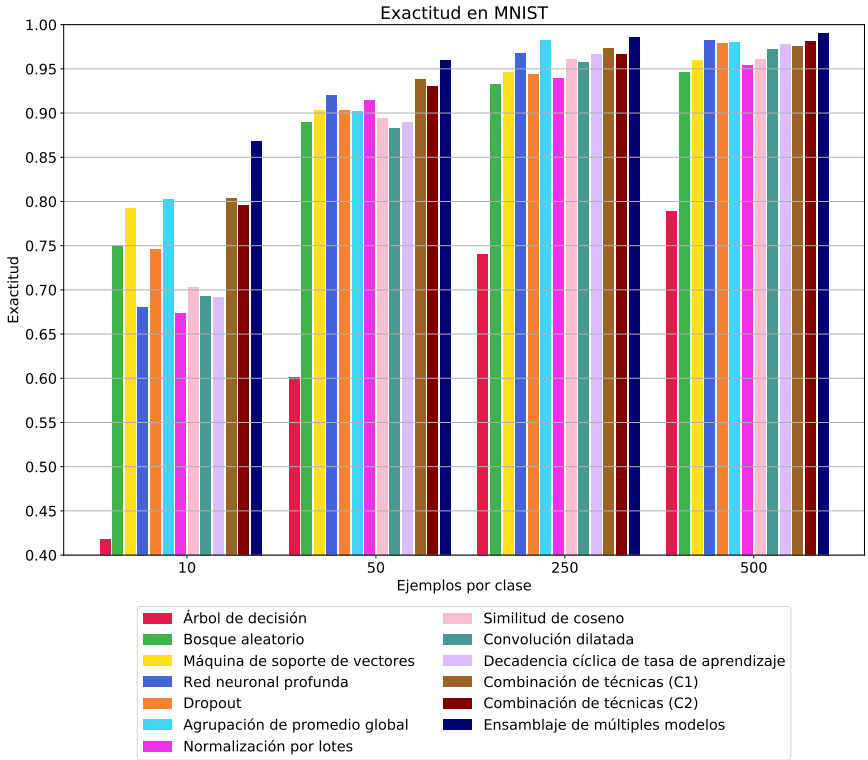

Figura 6.3: Gráfico de resultados de exactitud entre algoritmos evaluados sobre MNIST.

Precisión Los resultados agregados por precisión se presentan en la Tabla 6.2, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.4 se puede observar un gráfico de barras de la precisión obtenida por cada algoritmo y técnica.

Precisión				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,425 (-37,22 %)	0,602 (-34,64 %)	0,737 (-23,86 %)	0,786 (-19,96 %)
Bosque aleatorio	0,748 (10,49 %)	0,889 (-3,47 %)	0,934 (-3,51 %)	0,945 (-3,77 %)
Máquina de soporte de vectores	0,792 (16,99 %)	0,902 (-2,06 %)	0,946 (-2,27 %)	0,959 (-2,34 %)
Red Neuronal Profunda	0,677 (0,0 %)	0,921 (0,0 %)	0,968 (0,0 %)	0,982 (0,0 %)
Dropout	0,775 (14,48 %)	0,916 (-0,54 %)	0,974 (0,62 %)	0,98 (-0,2 %)
Agrupación Promedio Global	0,813 (20,09 %)	0,916 (-0,54 %)	0,982 (1,45 %)	0,98 (-0,2 %)
Normalización por lotes	0,712 (5,17 %)	0,916 (-0,54 %)	0,943 (-2,58 %)	0,956 (-2,65 %)
Similitud de coseno	0,715 (5,61 %)	0,898 (-2,5 %)	0,962 (-0,62 %)	0,962 (-2,04 %)
Convolución dilatada	0,693 (2,36 %)	0,882 (-4,23 %)	0,957 (-1,14 %)	0,972 (-1,02 %)
Decadencia cíclica de tasa de aprendizaje	0,701 (3,55 %)	0,89 (-3,37 %)	0,967 (-0,1 %)	0,978 (-0,41 %)
Combinación de técnicas (C1)	0,811 (19,79 %)	0,939 (1,95 %)	0,973 (0,52 %)	0,976 (-0,61 %)
Combinación de técnicas (C2)	0,811 (19,79 %)	0,93 (0,98 %)	0,967 (-0,1 %)	0,981 (-0,1 %)
Ensamblaje	0,875 (29,25 %)	0,96 (4,23 %)	0,986 (1,86 %)	0,99 (0,81 %)

Tabla 6.2: Comparación de resultados de precisión sobre MNIST.

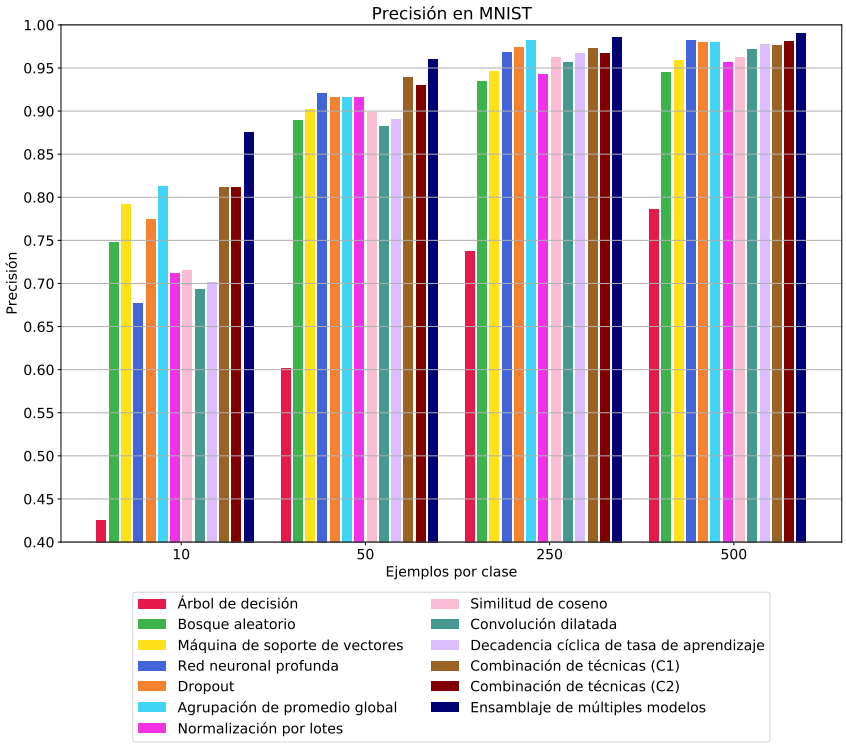

Figura 6.4: Gráfico de resultados de precisión entre algoritmos evaluados sobre MNIST.

Recall Los resultados agregados por recall se presentan en la Tabla 6.3, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.5 se puede observar un gráfico de barras de recall obtenido por cada algoritmo y técnica.

Recall				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,412 (-38,6 %)	0,597 (-35,04 %)	0,737 (-23,86 %)	0,787 (-19,78 %)
Bosque aleatorio	0,745 (11,03 %)	0,888 (-3,37 %)	0,934 (-3,51 %)	0,945 (-3,67 %)
Máquina de soporte de vectores	0,788 (17,44 %)	0,902 (-1,85 %)	0,946 (-2,27 %)	0,959 (-2,24 %)
Red Neuronal Profunda	0,671 (0,0 %)	0,919 (0,0 %)	0,968 (0,0 %)	0,981 (0,0 %)
Dropout	0,741 (10,43 %)	0,901 (-1,96 %)	0,974 (0,62 %)	0,979 (-0,2 %)
Agrupación Promedio Global	0,801 (19,37 %)	0,897 (-2,39 %)	0,982 (1,45 %)	0,98 (-0,1 %)
Normalización por lotes	0,671 (0,0 %)	0,912 (-0,76 %)	0,939 (-3,0 %)	0,955 (-2,65 %)
Similitud de coseno	0,698 (4,02 %)	0,892 (-2,94 %)	0,961 (-0,72 %)	0,961 (-2,04 %)
Convolución dilatada	0,686 (2,24 %)	0,88 (-4,24 %)	0,956 (-1,24 %)	0,972 (-0,92 %)
Decadencia cíclica de tasa de aprendizaje	0,687 (2,38 %)	0,889 (-3,26 %)	0,967 (-0,1 %)	0,978 (-0,31 %)
Combinación de técnicas (C1)	0,799 (19,08 %)	0,937 (1,96 %)	0,973 (0,52 %)	0,976 (-0,51 %)
Combinación de técnicas (C2)	0,791 (17,88 %)	0,929 (1,09 %)	0,967 (-0,1 %)	0,981 (0,0 %)
Ensamblaje	0,863 (28,61 %)	0,959 (4,35 %)	0,986 (1,86 %)	0,99 (0,92 %)

Tabla 6.3: Comparación de resultados de recall sobre MNIST.

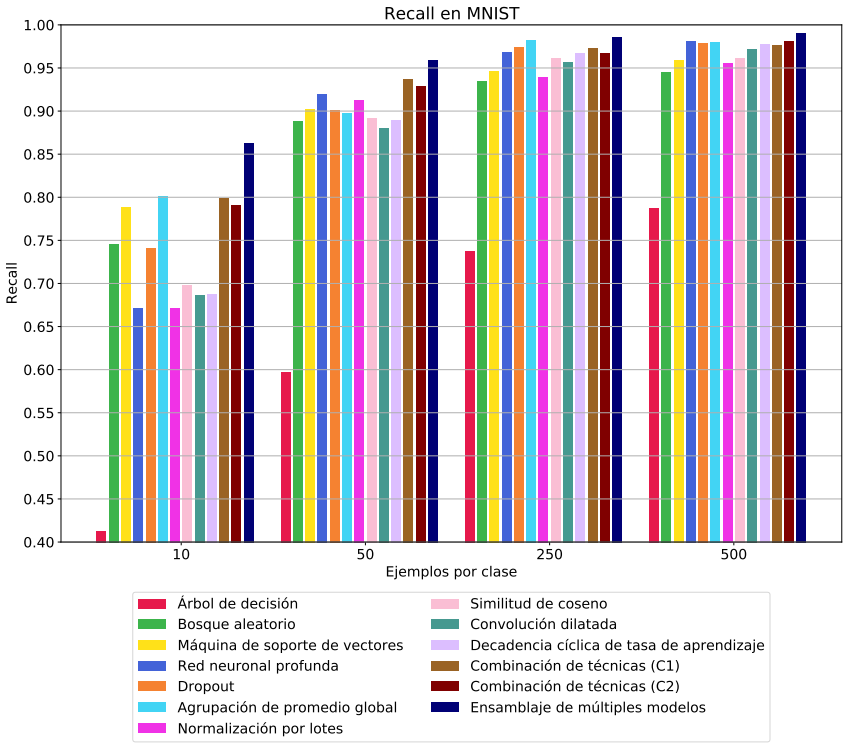

Figura 6.5: Gráfico de resultados de recall entre algoritmos evaluados sobre MNIST.

F1 Los resultados agregados por F1 se presentan en la Tabla 6.4, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.6 se puede observar un gráfico de barras de F1 obtenido por cada algoritmo y técnica.

F1				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,409 (-38,59 %)	0,596 (-35,15 %)	0,736 (-23,97 %)	0,786 (-19,88 %)
Bosque aleatorio	0,741 (11,26 %)	0,888 (-3,37 %)	0,933 (-3,62 %)	0,945 (-3,67 %)
Máquina de soporte de vectores	0,787 (18,17 %)	0,902 (-1,85 %)	0,946 (-2,27 %)	0,959 (-2,24 %)
Red Neuronal Profunda	0,666 (0,0 %)	0,919 (0,0 %)	0,968 (0,0 %)	0,981 (0,0 %)
Dropout	0,737 (10,66 %)	0,903 (-1,74 %)	0,974 (0,62 %)	0,979 (-0,2 %)
Agrupación Promedio Global	0,802 (20,42 %)	0,898 (-2,29 %)	0,982 (1,45 %)	0,98 (-0,1 %)
Normalización por lotes	0,676 (1,5 %)	0,913 (-0,65 %)	0,939 (-3,0 %)	0,955 (-2,65 %)
Similitud de coseno	0,699 (4,95 %)	0,893 (-2,83 %)	0,961 (-0,72 %)	0,961 (-2,04 %)
Convolución dilatada	0,683 (2,55 %)	0,881 (-4,13 %)	0,957 (-1,14 %)	0,972 (-0,92 %)
Decadencia cíclica de tasa de aprendizaje	0,683 (2,55 %)	0,889 (-3,26 %)	0,967 (-0,1 %)	0,978 (-0,31 %)
Combinación de técnicas (C1)	0,798 (19,82 %)	0,937 (1,96 %)	0,973 (0,52 %)	0,976 (-0,51 %)
Combinación de técnicas (C2)	0,791 (18,77 %)	0,929 (1,09 %)	0,967 (-0,1 %)	0,981 (0,0 %)
Ensamblaje	0,862 (29,43 %)	0,959 (4,35 %)	0,986 (1,86 %)	0,99 (0,92 %)

Tabla 6.4: Comparación de resultados de F1 sobre MNIST.

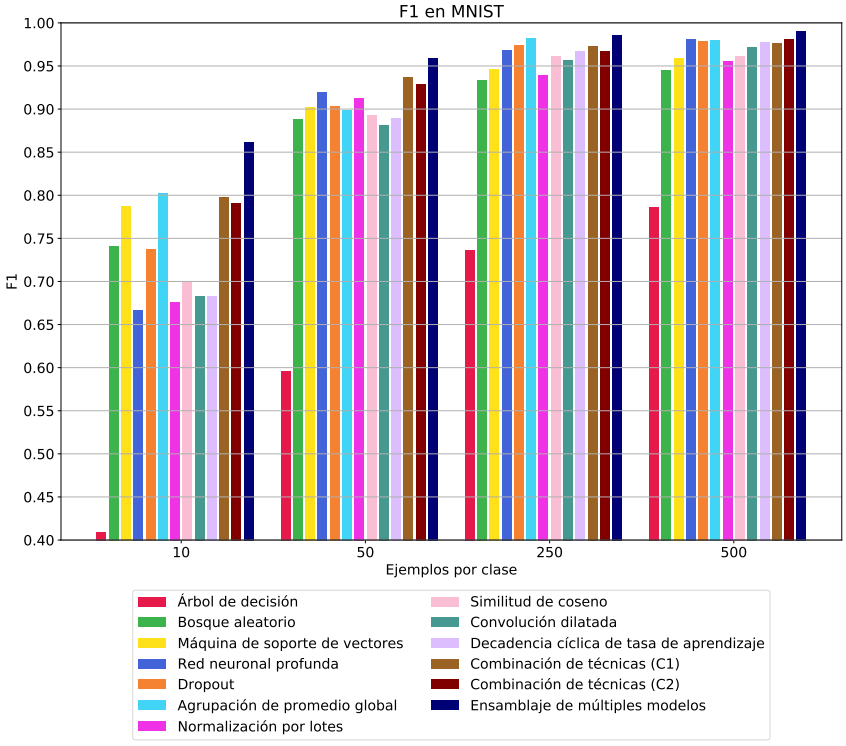

Figura 6.6: Gráfico de resultados de F1 entre algoritmos evaluados sobre MNIST.

Coeficiente de correlación de Matthews Los resultados agregados por MCC se presentan en la Tabla 6.5, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.7 se puede observar un gráfico de barras de MCC obtenido por cada algoritmo y técnica.

Coeficiente de correlación de Matthews				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,355 (-45,05 %)	0,557 (-38,86 %)	0,711 (-26,24 %)	0,766 (-21,84 %)
Bosque aleatorio	0,722 (11,76 %)	0,878 (-3,62 %)	0,927 (-3,84 %)	0,94 (-4,08 %)
Máquina de soporte de vectores	0,769 (19,04 %)	0,892 (-2,09 %)	0,94 (-2,49 %)	0,955 (-2,55 %)
Red Neuronal Profunda	0,646 (0,0 %)	0,911 (0,0 %)	0,964 (0,0 %)	0,98 (0,0 %)
Dropout	0,722 (11,76 %)	0,894 (-1,87 %)	0,971 (0,73 %)	0,977 (-0,31 %)
Agrupación Promedio Global	0,783 (21,21 %)	0,892 (-2,09 %)	0,98 (1,66 %)	0,978 (-0,2 %)
Normalización por lotes	0,642 (-0,62 %)	0,904 (-0,77 %)	0,934 (-3,11 %)	0,949 (-3,16 %)
Similitud de coseno	0,671 (3,87 %)	0,883 (-3,07 %)	0,957 (-0,73 %)	0,957 (-2,35 %)
Convolución dilatada	0,661 (2,32 %)	0,87 (-4,5 %)	0,952 (-1,24 %)	0,969 (-1,12 %)
Decadencia cíclica de tasa de aprendizaje	0,661 (2,32 %)	0,878 (-3,62 %)	0,964 (0,0 %)	0,976 (-0,41 %)
Combinación de técnicas (C1)	0,783 (21,21 %)	0,931 (2,2 %)	0,97 (0,62 %)	0,973 (-0,71 %)
Combinación de técnicas (C2)	0,775 (19,97 %)	0,922 (1,21 %)	0,963 (-0,1 %)	0,979 (-0,1 %)
Ensamblaje	0,854 (32,2 %)	0,955 (4,83 %)	0,985 (2,18 %)	0,989 (0,92 %)

Tabla 6.5: Comparación de resultados de coeficiente de correlación de Matthews sobre MNIST.

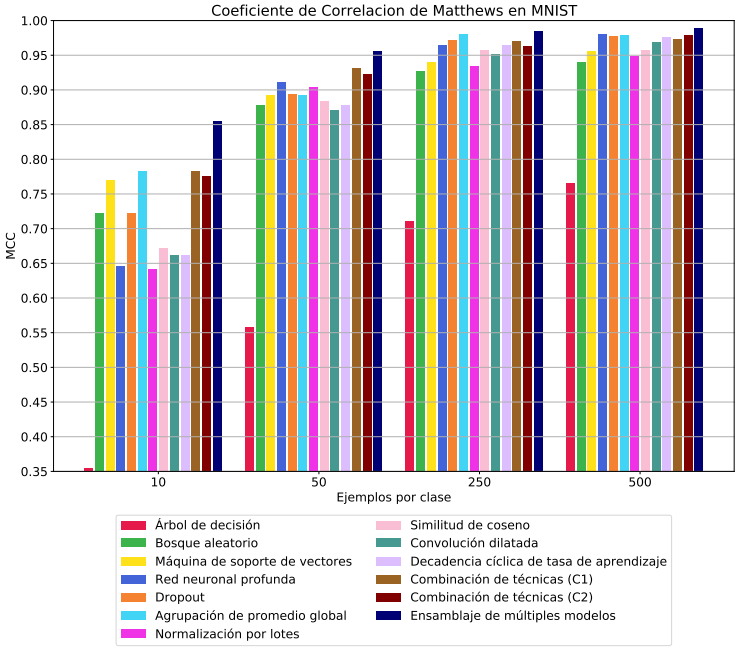

Figura 6.7: Gráfico de resultados de coeficiente de correlación de Matthews entre algoritmos evaluados sobre MNIST.

6.2.2. Subconjuntos de Fashion MNIST

Exactitud Los resultados agregados por exactitud se presentan en la Tabla 6.6, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.8 se puede observar un gráfico de barras de la exactitud obtenida por cada algoritmo y técnica.

Exactitud				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,463 (-25,8 %)	0,649 (-14,83 %)	0,729 (-9,1 %)	0,756 (-11,48 %)
Bosque aleatorio	0,675 (8,17 %)	0,778 (2,1 %)	0,822 (2,49 %)	0,837 (-1,99 %)
Máquina de soporte de vectores	0,685 (9,78 %)	0,798 (4,72 %)	0,837 (4,36 %)	0,851 (-0,35 %)
Red Neuronal Profunda	0,624 (0,0 %)	0,762 (0,0 %)	0,802 (0,0 %)	0,854 (0,0 %)
Dropout	0,617 (-1,12 %)	0,718 (-5,77 %)	0,844 (5,24 %)	0,87 (1,87 %)
Agrupación Promedio Global	0,683 (9,46 %)	0,792 (3,94 %)	0,854 (6,48 %)	0,846 (-0,94 %)
Normalización por lotes	0,613 (-1,76 %)	0,781 (2,49 %)	0,816 (1,75 %)	0,866 (1,41 %)
Similitud de coseno	0,631 (1,12 %)	0,743 (-2,49 %)	0,803 (0,12 %)	0,855 (0,12 %)
Convolución dilatada	0,653 (4,65 %)	0,78 (2,36 %)	0,837 (4,36 %)	0,856 (0,23 %)
Decadencia cíclica de tasa de aprendizaje	0,63 (0,96 %)	0,738 (-3,15 %)	0,827 (3,12 %)	0,851 (-0,35 %)
Combinación de técnicas (C1)	0,676 (8,33 %)	0,781 (2,49 %)	0,813 (1,37 %)	0,843 (-1,29 %)
Combinación de técnicas (C2)	0,655 (4,97 %)	0,784 (2,89 %)	0,834 (3,99 %)	0,843 (-1,29 %)
Ensamblaje	0,702 (12,5 %)	0,819 (7,48 %)	0,874 (8,98 %)	0,89 (4,22 %)

Tabla 6.6: Comparación de resultados de exactitud sobre Fashion MNIST.

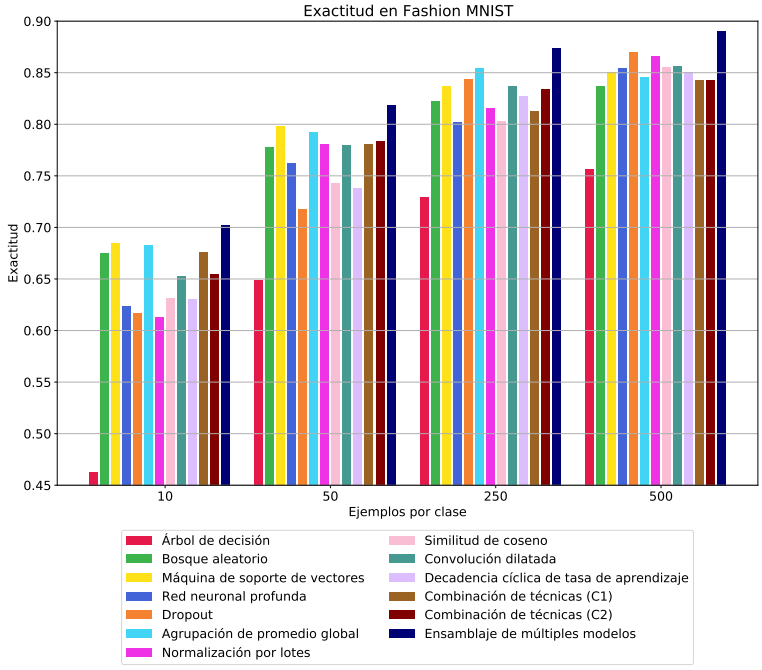

Figura 6.8: Gráfico de resultados de exactitud entre algoritmos evaluados sobre Fashion MNIST.

Precisión Los resultados agregados por precisión se presentan en la Tabla 6.7, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.9 se puede observar un gráfico de barras de la precisión obtenida por cada algoritmo y técnica.

Precisión				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,484 (-23,54 %)	0,618 (-20,77 %)	0,731 (-12,77 %)	0,755 (-11,59 %)
Bosque aleatorio	0,674 (6,48 %)	0,774 (-0,77 %)	0,82 (-2,15 %)	0,835 (-2,22 %)
Máquina de soporte de vectores	0,685 (8,21 %)	0,799 (2,44 %)	0,837 (-0,12 %)	0,85 (-0,47 %)
Red Neuronal Profunda	0,633 (0,0 %)	0,78 (0,0 %)	0,838 (0,0 %)	0,854 (0,0 %)
Dropout	0,619 (-2,21 %)	0,732 (-6,15 %)	0,841 (0,36 %)	0,872 (2,11 %)
Agrupación Promedio Global	0,681 (7,58 %)	0,796 (2,05 %)	0,857 (2,27 %)	0,858 (0,47 %)
Normalización por lotes	0,63 (-0,47 %)	0,794 (1,79 %)	0,828 (-1,19 %)	0,868 (1,64 %)
Similitud de coseno	0,643 (1,58 %)	0,746 (-4,36 %)	0,811 (-3,22 %)	0,862 (0,94 %)
Convolución dilatada	0,639 (0,95 %)	0,78 (0,0 %)	0,84 (0,24 %)	0,854 (0,0 %)
Decadencia cíclica de tasa de aprendizaje	0,633 (0,0 %)	0,74 (-5,13 %)	0,835 (-0,36 %)	0,851 (-0,35 %)
Combinación de técnicas (C1)	0,683 (7,9 %)	0,789 (1,15 %)	0,822 (-1,91 %)	0,851 (-0,35 %)
Combinación de técnicas (C2)	0,683 (7,9 %)	0,803 (2,95 %)	0,843 (0,6 %)	0,854 (0,0 %)
Ensamblaje	0,699 (10,43 %)	0,814 (4,36 %)	0,874 (4,3 %)	0,89 (4,22 %)

Tabla 6.7: Comparación de resultados de precisión sobre Fashion MNIST.

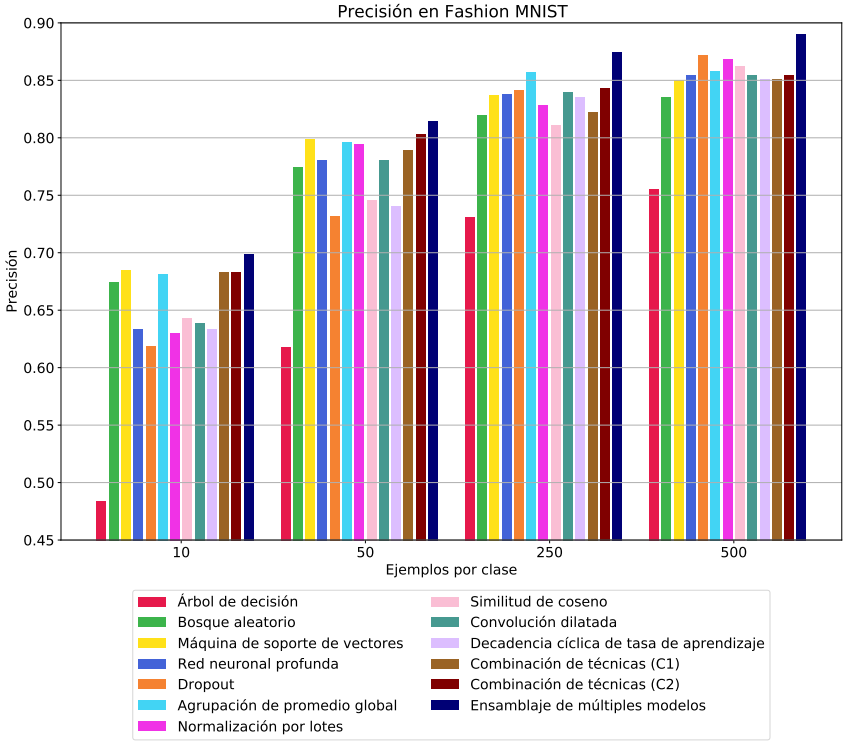

Figura 6.9: Gráfico de resultados de precisión entre algoritmos evaluados sobre Fashion MNIST.

Recall Los resultados agregados por recall se presentan en la Tabla 6.8, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.10 se puede observar un gráfico de barras de recall obtenido por cada algoritmo y técnica.

Recall				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,463 (-25,8 %)	0,649 (-14,83 %)	0,729 (-9,1 %)	0,756 (-11,48 %)
Bosque aleatorio	0,675 (8,17 %)	0,778 (2,1 %)	0,822 (2,49 %)	0,837 (-1,99 %)
Máquina de soporte de vectores	0,685 (9,78 %)	0,798 (4,72 %)	0,837 (4,36 %)	0,851 (-0,35 %)
Red Neuronal Profunda	0,624 (0,0 %)	0,762 (0,0 %)	0,802 (0,0 %)	0,854 (0,0 %)
Dropout	0,617 (-1,12 %)	0,718 (-5,77 %)	0,844 (5,24 %)	0,87 (1,87 %)
Agrupación Promedio Global	0,683 (9,46 %)	0,792 (3,94 %)	0,854 (6,48 %)	0,846 (-0,94 %)
Normalización por lotes	0,613 (-1,76 %)	0,781 (2,49 %)	0,816 (1,75 %)	0,866 (1,41 %)
Similitud de coseno	0,631 (1,12 %)	0,743 (-2,49 %)	0,803 (0,12 %)	0,855 (0,12 %)
Convolución dilatada	0,653 (4,65 %)	0,78 (2,36 %)	0,837 (4,36 %)	0,856 (0,23 %)
Decadencia cíclica de tasa de aprendizaje	0,63 (0,96 %)	0,738 (-3,15 %)	0,827 (3,12 %)	0,851 (-0,35 %)
Combinación de técnicas (C1)	0,676 (8,33 %)	0,781 (2,49 %)	0,813 (1,37 %)	0,843 (-1,29 %)
Combinación de técnicas (C2)	0,655 (4,97 %)	0,784 (2,89 %)	0,834 (3,99 %)	0,843 (-1,29 %)
Ensamblaje	0,702 (12,5 %)	0,819 (7,48 %)	0,874 (8,98 %)	0,89 (4,22 %)

Tabla 6.8: Comparación de resultados de recall sobre Fashion MNIST.

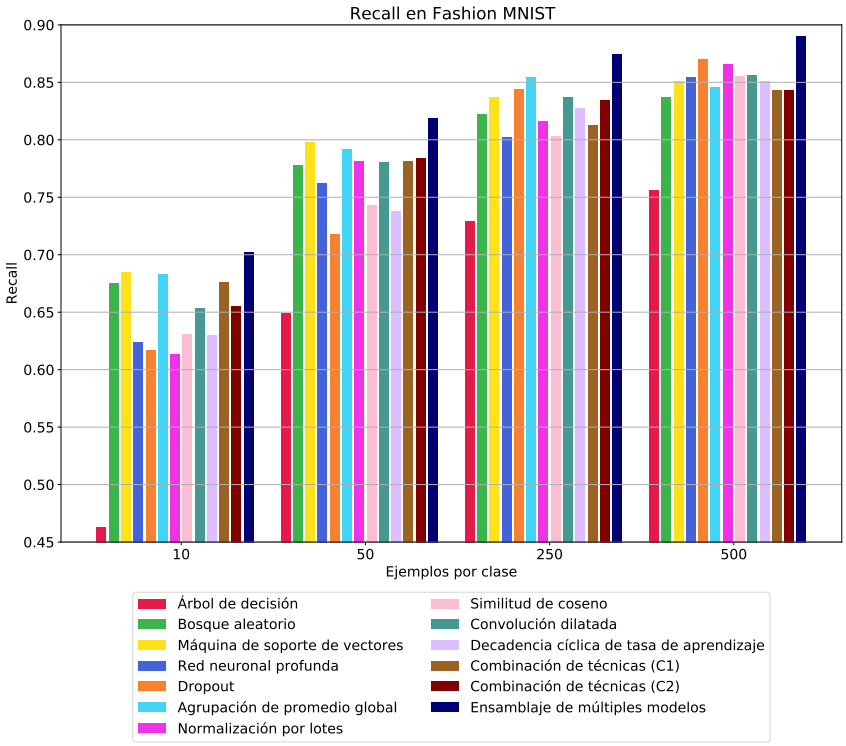

Figura 6.10: Gráfico de resultados de recall entre algoritmos evaluados sobre Fashion MNIST.

F1 Los resultados agregados por F1 se presentan en la Tabla 6.9, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.11 se puede observar un gráfico de barras de F1 obtenido por cada algoritmo y técnica.

F1				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,462 (-25,72 %)	0,618 (-18,04 %)	0,728 (-9,34 %)	0,754 (-11,71 %)
Bosque aleatorio	0,673 (8,2 %)	0,775 (2,79 %)	0,82 (2,12 %)	0,834 (-2,34 %)
Máquina de soporte de vectores	0,682 (9,65 %)	0,798 (5,84 %)	0,837 (4,23 %)	0,85 (-0,47 %)
Red Neuronal Profunda	0,622 (0,0 %)	0,754 (0,0 %)	0,803 (0,0 %)	0,854 (0,0 %)
Dropout	0,592 (-4,82 %)	0,718 (-4,77 %)	0,841 (4,73 %)	0,87 (1,87 %)
Agrupación Promedio Global	0,679 (9,16 %)	0,79 (4,77 %)	0,853 (6,23 %)	0,845 (-1,05 %)
Normalización por lotes	0,611 (-1,77 %)	0,77 (2,12 %)	0,815 (1,49 %)	0,866 (1,41 %)
Similitud de coseno	0,629 (1,13 %)	0,735 (-2,52 %)	0,803 (0,0 %)	0,856 (0,23 %)
Convolución dilatada	0,638 (2,57 %)	0,777 (3,05 %)	0,836 (4,11 %)	0,854 (0,0 %)
Decadencia cíclica de tasa de aprendizaje	0,629 (1,13 %)	0,734 (-2,65 %)	0,829 (3,24 %)	0,85 (-0,47 %)
Combinación de técnicas (C1)	0,678 (9,0 %)	0,783 (3,85 %)	0,812 (1,12 %)	0,845 (-1,05 %)
Combinación de técnicas (C2)	0,644 (3,54 %)	0,786 (4,24 %)	0,833 (3,74 %)	0,845 (-1,05 %)
Ensamblaje	0,695 (11,74 %)	0,815 (8,09 %)	0,874 (8,84 %)	0,89 (4,22 %)

Tabla 6.9: Comparación de resultados de F1 sobre Fashion MNIST.

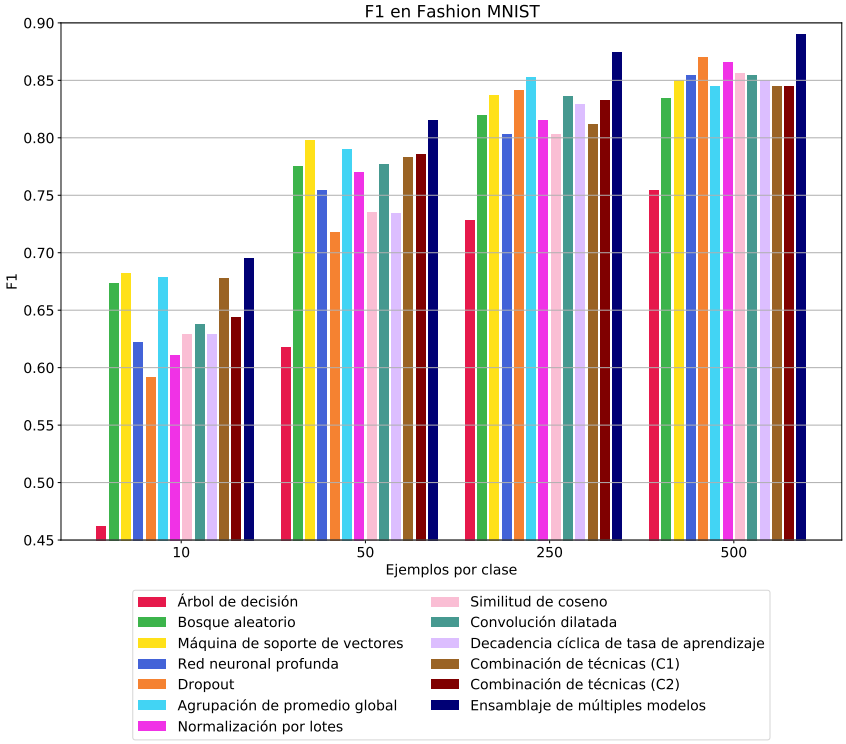

Figura 6.11: Gráfico de resultados de F1 entre algoritmos evaluados sobre Fashion MNIST.

Coeficiente de correlación de Matthews Los resultados agregados por MCC se presentan en la Tabla 6.10, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.12 se puede observar un gráfico de barras de MCC obtenido por cada algoritmo y técnica.

Coeficiente de correlación de Matthews				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,406 (-30,48 %)	0,617 (-16,51 %)	0,699 (-10,84 %)	0,729 (-13,01 %)
Bosque aleatorio	0,639 (9,42 %)	0,754 (2,03 %)	0,803 (2,42 %)	0,819 (-2,27 %)
Máquina de soporte de vectores	0,65 (11,3 %)	0,775 (4,87 %)	0,819 (4,46 %)	0,835 (-0,36 %)
Red Neuronal Profunda	0,584 (0,0 %)	0,739 (0,0 %)	0,784 (0,0 %)	0,838 (0,0 %)
Dropout	0,58 (-0,68 %)	0,688 (-6,9 %)	0,827 (5,48 %)	0,856 (2,15 %)
Agrupación Promedio Global	0,649 (11,13 %)	0,771 (4,33 %)	0,838 (6,89 %)	0,83 (-0,95 %)
Normalización por lotes	0,573 (-1,88 %)	0,762 (3,11 %)	0,797 (1,66 %)	0,851 (1,55 %)
Similitud de coseno	0,591 (1,2 %)	0,717 (-2,98 %)	0,782 (-0,26 %)	0,839 (0,12 %)
Convolución dilatada	0,617 (5,65 %)	0,757 (2,44 %)	0,819 (4,46 %)	0,84 (0,24 %)
Decadencia cíclica de tasa de aprendizaje	0,589 (0,86 %)	0,71 (-3,92 %)	0,808 (3,06 %)	0,834 (-0,48 %)
Combinación de técnicas (C1)	0,641 (9,76 %)	0,757 (2,44 %)	0,793 (1,15 %)	0,827 (-1,31 %)
Combinación de técnicas (C2)	0,623 (6,68 %)	0,761 (2,98 %)	0,817 (4,21 %)	0,827 (-1,31 %)
Ensamblaje	0,67 (14,73 %)	0,799 (8,12 %)	0,86 (9,69 %)	0,878 (4,77 %)

Tabla 6.10: Comparación de resultados de coeficiente de correlación de Matthews sobre Fashion MNIST.

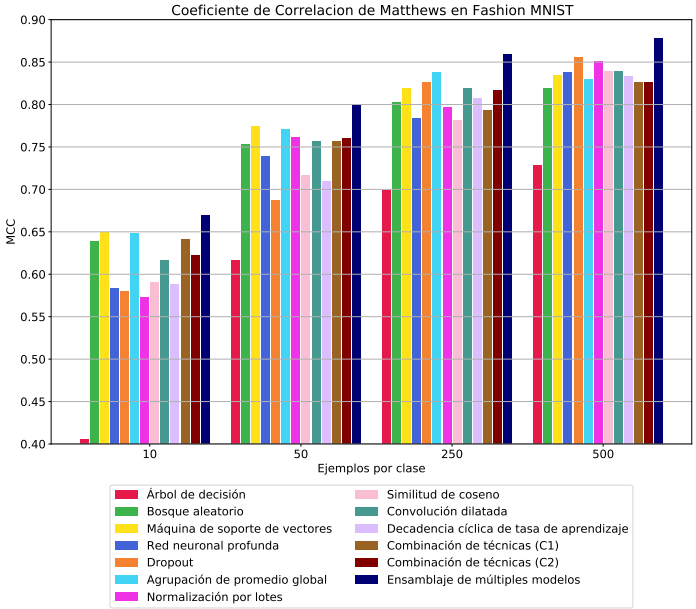

Figura 6.12: Gráfico de resultados de coeficiente de correlación de Matthews entre algoritmos evaluados sobre Fashion MNIST.

6.2.3. Subconjuntos de CIFAR-10

Exactitud Los resultados agregados por exactitud se presentan en la Tabla 6.11, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.13 se puede observar un gráfico de barras de la exactitud obtenida por cada algoritmo y técnica.

Exactitud				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,164 (-42,46 %)	0,192 (-46,22 %)	0,229 (-53,17 %)	0,252 (-56,17 %)
Bosque aleatorio	0,284 (-0,35 %)	0,334 (-6,44 %)	0,403 (-17,59 %)	0,425 (-26,09 %)
Máquina de soporte de vectores	0,262 (-8,07 %)	0,339 (-5,04 %)	0,422 (-13,7 %)	0,456 (-20,7 %)
Red Neuronal Profunda	0,285 (0,0 %)	0,357 (0,0 %)	0,489 (0,0 %)	0,575 (0,0 %)
Dropout	0,294 (3,16 %)	0,389 (8,96 %)	0,524 (7,16 %)	0,604 (5,04 %)
Agrupación Promedio Global	0,293 (2,81 %)	0,39 (9,24 %)	0,498 (1,84 %)	0,59 (2,61 %)
Normalización por lotes	0,295 (3,51 %)	0,41 (14,85 %)	0,512 (4,7 %)	0,569 (-1,04 %)
Similitud de coseno	0,269 (-5,61 %)	0,375 (5,04 %)	0,508 (3,89 %)	0,573 (-0,35 %)
Convolución dilatada	0,262 (-8,07 %)	0,35 (-1,96 %)	0,473 (-3,27 %)	0,531 (-7,65 %)
Decadencia cíclica de tasa de aprendizaje	0,288 (1,05 %)	0,36 (0,84 %)	0,507 (3,68 %)	0,562 (-2,26 %)
Combinación de técnicas (C1)	0,291 (2,11 %)	0,385 (7,84 %)	0,469 (-4,09 %)	0,544 (-5,39 %)
Combinación de técnicas (C2)	0,378 (32,63 %)	0,584 (63,59 %)	0,724 (48,06 %)	0,762 (32,52 %)
Ensamblaje	0,319 (11,93 %)	0,43 (20,45 %)	0,56 (14,52 %)	0,642 (11,65 %)

Tabla 6.11: Comparación de resultados de exactitud sobre CIFAR-10.

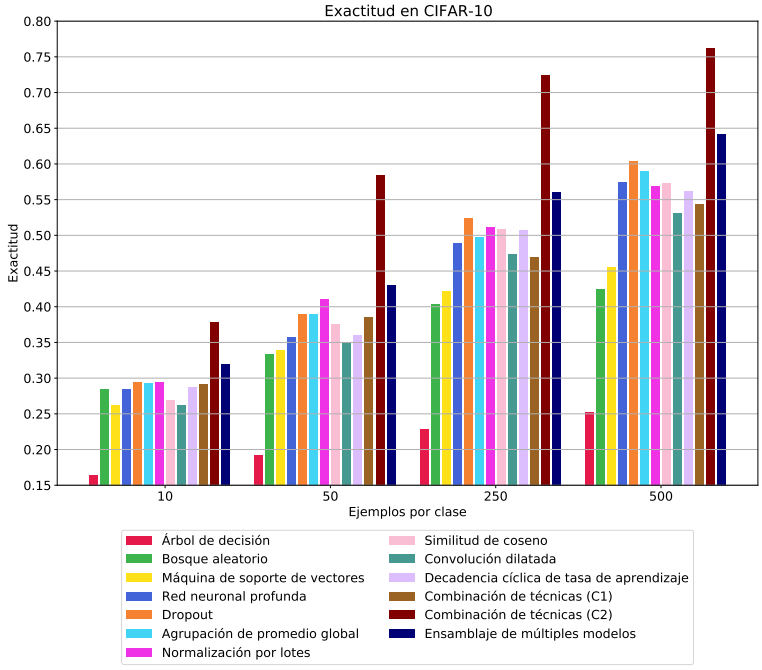

Figura 6.13: Gráfico de resultados de exactitud entre algoritmos evaluados sobre CIFAR-10.

Precisión Los resultados agregados por precisión se presentan en la Tabla 6.12, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.14 se puede observar un gráfico de barras de la precisión obtenida por cada algoritmo y técnica.

Precisión				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,184 (-35,44 %)	0,196 (-48,01 %)	0,238 (-52,78 %)	0,254 (-55,52 %)
Bosque aleatorio	0,294 (3,16 %)	0,331 (-12,2 %)	0,396 (-21,43 %)	0,418 (-26,8 %)
Máquina de soporte de vectores	0,279 (-2,11 %)	0,348 (-7,69 %)	0,42 (-16,67 %)	0,456 (-20,14 %)
Red Neuronal Profunda	0,285 (0,0 %)	0,377 (0,0 %)	0,504 (0,0 %)	0,571 (0,0 %)
Dropout	0,3 (5,26 %)	0,411 (9,02 %)	0,526 (4,37 %)	0,604 (5,78 %)
Agrupación Promedio Global	0,283 (-0,7 %)	0,391 (3,71 %)	0,487 (-3,37 %)	0,609 (6,65 %)
Normalización por lotes	0,302 (5,96 %)	0,417 (10,61 %)	0,519 (2,98 %)	0,585 (2,45 %)
Similitud de coseno	0,288 (1,05 %)	0,395 (4,77 %)	0,503 (-0,2 %)	0,58 (1,58 %)
Convolución dilatada	0,272 (-4,56 %)	0,353 (-6,37 %)	0,478 (-5,16 %)	0,531 (-7,01 %)
Decadencia cíclica de tasa de aprendizaje	0,282 (-1,05 %)	0,353 (-6,37 %)	0,498 (-1,19 %)	0,563 (-1,4 %)
Combinación de técnicas (C1)	0,302 (5,96 %)	0,394 (4,51 %)	0,475 (-5,75 %)	0,541 (-5,25 %)
Combinación de técnicas (C2)	0,39 (36,84 %)	0,617 (63,66 %)	0,728 (44,44 %)	0,768 (34,5 %)
Ensamblaje	0,315 (10,53 %)	0,432 (14,59 %)	0,558 (10,71 %)	0,637 (11,56 %)

Tabla 6.12: Comparación de resultados de precisión sobre CIFAR-10.

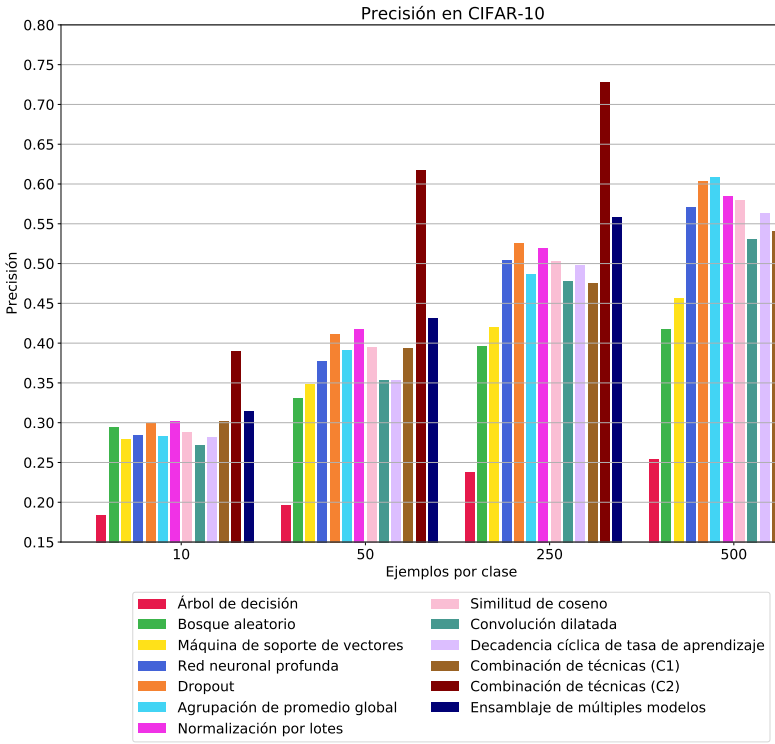

Figura 6.14: Gráfico de resultados de precisión entre algoritmos evaluados sobre CIFAR-10.

Recall Los resultados agregados por recall se presentan en la Tabla 6.13, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.15 se puede observar un gráfico de barras de recall obtenido por cada algoritmo y técnica.

Recall				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,164 (-42,46 %)	0,192 (-46,22 %)	0,229 (-53,17 %)	0,252 (-56,17 %)
Bosque aleatorio	0,284 (-0,35 %)	0,334 (-6,44 %)	0,403 (-17,59 %)	0,425 (-26,09 %)
Máquina de soporte de vectores	0,262 (-8,07 %)	0,339 (-5,04 %)	0,422 (-13,7 %)	0,456 (-20,7 %)
Red Neuronal Profunda	0,285 (0,0 %)	0,357 (0,0 %)	0,489 (0,0 %)	0,575 (0,0 %)
Dropout	0,294 (3,16 %)	0,389 (8,96 %)	0,524 (7,16 %)	0,604 (5,04 %)
Agrupación Promedio Global	0,293 (2,81 %)	0,39 (9,24 %)	0,498 (1,84 %)	0,59 (2,61 %)
Normalización por lotes	0,295 (3,51 %)	0,41 (14,85 %)	0,512 (4,7 %)	0,569 (-1,04 %)
Similitud de coseno	0,269 (-5,61 %)	0,375 (5,04 %)	0,508 (3,89 %)	0,573 (-0,35 %)
Convolución dilatada	0,262 (-8,07 %)	0,35 (-1,96 %)	0,473 (-3,27 %)	0,531 (-7,65 %)
Decadencia cíclica de tasa de aprendizaje	0,288 (1,05 %)	0,36 (0,84 %)	0,507 (3,68 %)	0,562 (-2,26 %)
Combinación de técnicas (C1)	0,291 (2,11 %)	0,385 (7,84 %)	0,469 (-4,09 %)	0,544 (-5,39 %)
Combinación de técnicas (C2)	0,378 (32,63 %)	0,584 (63,59 %)	0,724 (48,06 %)	0,762 (32,52 %)
Ensamblaje	0,319 (11,93 %)	0,43 (20,45 %)	0,567 (15,95 %)	0,642 (11,65 %)

Tabla 6.13: Comparación de resultados de recall sobre CIFAR-10.

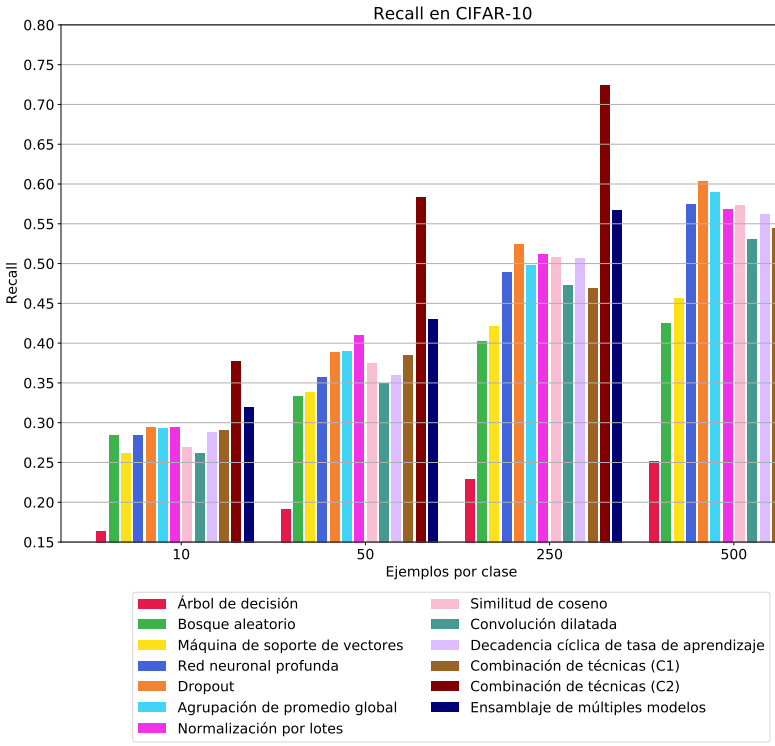

Figura 6.15: Gráfico de resultados de recall entre algoritmos evaluados sobre CIFAR-10.

F1 Los resultados agregados por F1 se presentan en la Tabla 6.14, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.16 se puede observar un gráfico de barras de F1 obtenido por cada algoritmo y técnica.

F1				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,165 (-40,65 %)	0,193 (-45,94 %)	0,217 (-55,07 %)	0,246 (-56,77 %)
Bosque aleatorio	0,272 (-2,16 %)	0,329 (-7,84 %)	0,396 (-18,01 %)	0,417 (-26,71 %)
Máquina de soporte de vectores	0,262 (-5,76 %)	0,34 (-4,76 %)	0,419 (-13,25 %)	0,455 (-20,04 %)
Red Neuronal Profunda	0,278 (0,0 %)	0,357 (0,0 %)	0,483 (0,0 %)	0,569 (0,0 %)
Dropout	0,292 (5,04 %)	0,393 (10,08 %)	0,519 (7,45 %)	0,601 (5,62 %)
Agrupación Promedio Global	0,277 (-0,36 %)	0,371 (3,92 %)	0,475 (-1,66 %)	0,59 (3,69 %)
Normalización por lotes	0,291 (4,68 %)	0,41 (14,85 %)	0,507 (4,97 %)	0,571 (0,35 %)
Similitud de coseno	0,262 (-5,76 %)	0,373 (4,48 %)	0,504 (4,35 %)	0,569 (0,0 %)
Convolución dilatada	0,254 (-8,63 %)	0,341 (-4,48 %)	0,469 (-2,9 %)	0,524 (-7,91 %)
Decadencia cíclica de tasa de aprendizaje	0,27 (-2,88 %)	0,349 (-2,24 %)	0,5 (3,52 %)	0,559 (-1,76 %)
Combinación de técnicas (C1)	0,293 (5,4 %)	0,386 (8,12 %)	0,466 (-3,52 %)	0,54 (-5,1 %)
Combinación de técnicas (C2)	0,351 (26,26 %)	0,574 (60,78 %)	0,717 (48,45 %)	0,758 (33,22 %)
Ensamblaje	0,31 (11,51 %)	0,426 (19,33 %)	0,56 (15,94 %)	0,639 (12,3 %)

Tabla 6.14: Comparación de resultados de F1 sobre CIFAR-10.

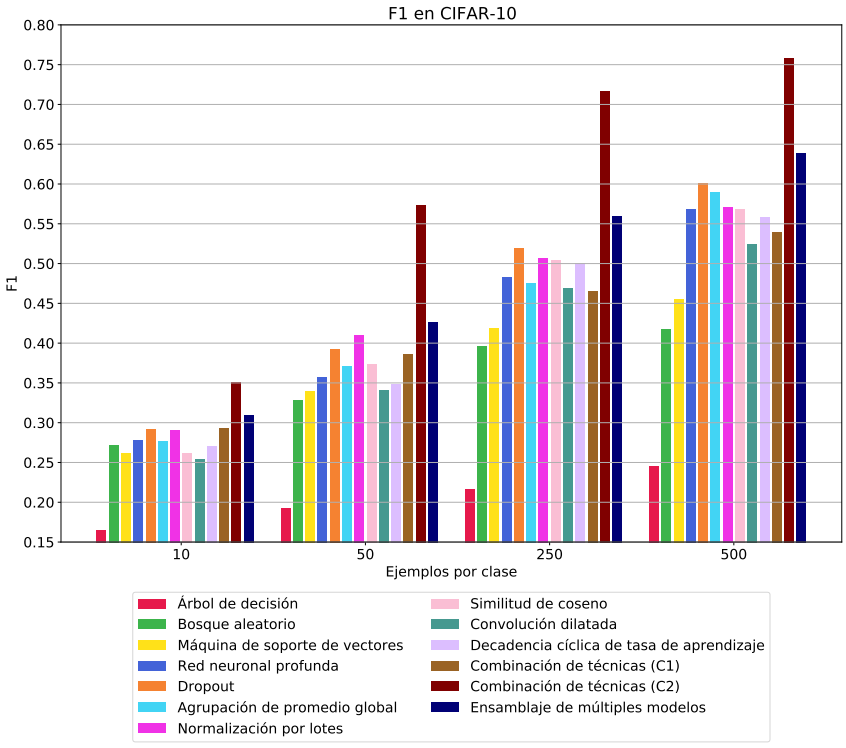

Figura 6.16: Gráfico de resultados de F1 entre algoritmos evaluados sobre CIFAR-10.

Coeficiente de correlación de Matthews Los resultados agregados por MCC se presentan en la Tabla 6.15, los valores entre paréntesis muestran el incremento respecto de la red neuronal sin técnicas. En la Figura 6.17 se puede observar un gráfico de barras de MCC obtenido por cada algoritmo y técnica.

Coeficiente de correlación de Matthews				
Ejemplos por clase	10	50	250	500
Árbol de decisión	0,071 (-65,53 %)	0,102 (-64,58 %)	0,146 (-66,44 %)	0,17 (-67,8 %)
Bosque aleatorio	0,207 (0,49 %)	0,261 (-9,38 %)	0,337 (-22,53 %)	0,362 (-31,44 %)
Máquina de soporte de vectores	0,181 (-12,14 %)	0,266 (-7,64 %)	0,357 (-17,93 %)	0,396 (-25,0 %)
Red Neuronal Profunda	0,206 (0,0 %)	0,288 (0,0 %)	0,435 (0,0 %)	0,528 (0,0 %)
Dropout	0,216 (4,85 %)	0,322 (11,81 %)	0,473 (8,74 %)	0,561 (6,25 %)
Agrupación Promedio Global	0,217 (5,34 %)	0,326 (13,19 %)	0,445 (2,3 %)	0,546 (3,41 %)
Normalización por lotes	0,217 (5,34 %)	0,345 (19,79 %)	0,46 (5,75 %)	0,522 (-1,14 %)
Similitud de coseno	0,191 (-7,28 %)	0,307 (6,6 %)	0,454 (4,37 %)	0,528 (0,0 %)
Convolución dilatada	0,184 (-10,68 %)	0,28 (-2,78 %)	0,415 (-4,6 %)	0,48 (-9,09 %)
Decadencia cíclica de tasa de aprendizaje	0,212 (2,91 %)	0,291 (1,04 %)	0,453 (4,14 %)	0,514 (-2,65 %)
Combinación de técnicas (C1)	0,212 (2,91 %)	0,317 (10,07 %)	0,412 (-5,29 %)	0,494 (-6,44 %)
Combinación de técnicas (C2)	0,315 (52,91 %)	0,544 (88,89 %)	0,695 (59,77 %)	0,738 (39,77 %)
Ensamblaje	0,245 (18,93 %)	0,368 (27,78 %)	0,519 (19,31 %)	0,603 (14,2 %)

Tabla 6.15: Comparación de resultados de coeficiente de correlación de Matthews sobre CIFAR-10.

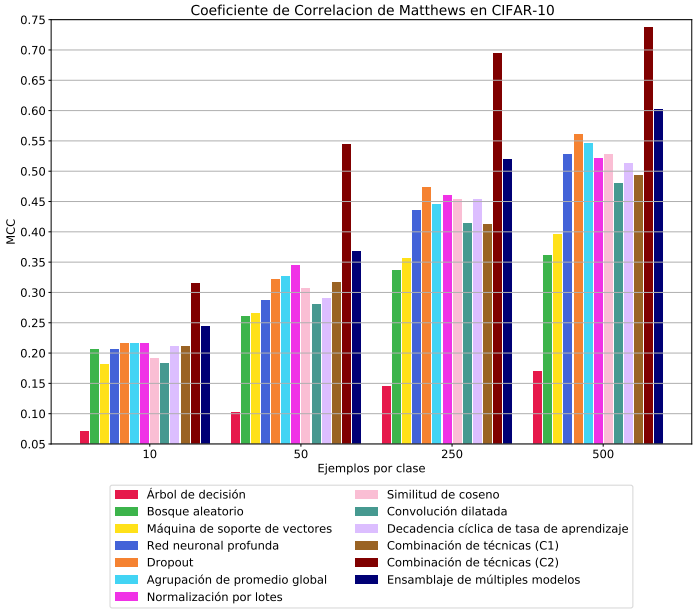

Figura 6.17: Gráfico de resultados de coeficiente de correlación de Matthews entre algoritmos evaluados sobre CIFAR-10.

6.2.4. Discusión de los resultados

En esta tesis, principalmente encontramos que los resultados son bastante dispersos, es decir, una técnica que se desempeña bien en un conjunto de datos, no necesariamente se va a desempeñar bien en otro conjunto de datos diferente, pero si existe una técnica que obtiene un desempeño mayor que la red neuronal base para cada conjunto de datos. Por ejemplo, el uso de la función de costo Similitud de Coseno, incrementa la exactitud en 3,38 % para el subconjunto de 10 imágenes por clase de MNIST, pero muestra una reducción de -5,61 % en el subconjunto del mismo tamaño de CIFAR-10.

Para los subconjuntos de MNIST y Fashion MNIST, el Ensamblaje de Múltiples Modelos con todas las redes neuronales ha sido la técnica que mejores resultados ha entregado para todos los tamaños de subconjunto. Además, ha mostrado los mayores incrementos respecto de la red neuronal base. Por otra parte, para el conjunto CIFAR-10 se observa que el Ensamblaje de Múltiples Modelos mejora los resultados respecto de la mayoría de los algoritmos pero es superado por la Combinación de técnicas C2. Se piensa que esto se debe a que el Ensamblaje de Múltiples Modelos opera como un promedio de los resultados, por lo que el resultado promedio es menor que el resultado más alto.

Respecto de la Combinación de técnicas C2, los resultados de CIFAR-10 muestran un aumento significativo en las métricas obtenidas, por ejemplo, el coef. de correlación de Matthews muestra un aumento entre 39 % y 88 %. Por lo tanto, se observa que la técnica de transferencia de aprendizaje es capaz de ayudar a compensar la limitación en la cantidad de datos de entrenamiento, tal como es indicado en el trabajo de Rajpurkar et al. (2020). Sin embargo, esto no sucede en los conjuntos de datos MNIST y Fashion MNIST, donde la Combinación de técnicas C2 presenta resultados menores que la Combinación de técnicas C1. Esto se debe a que los datos del conjunto de datos CIFAR-10, a diferencia de los de los conjuntos de datos MNIST y Fashion MNIST, son de un dominio cercano a los del conjunto ImagenNet, por lo tanto las técnicas de Transferencia de Aprendizaje sumado a la resistencia a las variaciones que aporta la técnica de Aumento de datos tienen un alto efecto en el desempeño obtenido. No siendo este el caso para los experimentos sobre los conjuntos de datos MNIST y Fashion MNIST.

Respecto de la técnica de Dropout, utilizada por los trabajos de Brigato y Iocchi (2020); Lu y Li (2018); Pasupa y Sunhem (2016); Yang et al. (2018), logra reducir el sobreajuste (la diferencia entre el costo de entrenamiento y el costo de prueba) de la red neuronal profunda, esto se puede observar al comparar los gráficos de pérdida/iteraciones sobre todo para los subconjuntos de 50 o más imágenes por clase.

El uso de la técnica de Agrupación de Promedio Global, por sí sola, sobre la red neuronal base muestra mejoras en el coef. de correlación de Matthews obtenido de hasta el 21,21 % y reducciones solo de -2,09 %, obteniendo los mayores incrementos en los subconjuntos más pequeños. Además, como indica el trabajo de Zhou et al. (2019), se comprueba que la aplicación de la capa de Agrupación de promedio global reduce la cantidad de parámetros de la red, debido a la eliminación de las capas densas.

En el trabajo de Barz y Denzler (2019) se establece que el uso de la función de costo Similitud de Coseno mejora las métricas obtenidas cuando se utiliza un conjunto pequeño de datos al compararse con la función de costo Entropía cruzada categórica. Sin embargo, los resultados obtenidos en esta tesis no permiten ratificar lo indicado, esto se debe a que en los conjuntos MNIST y Fashion MNIST se observa un aumento al aplicar esta técnica en los subconjuntos de 10 imágenes por clase pero una reducción en el resto de subconjuntos y para los subconjuntos de CIFAR-10 se observa lo contrario, una reducción en el subconjunto de 10 imágenes por clase y un aumento en el resto de subconjuntos.

Respecto de los algoritmos de aprendizaje clásicos, se destaca la Máquina de Soporte de Vectores, que obtiene resultados competitivos con las técnicas de aprendizaje profundo sobre todo en los subconjuntos de 10 imágenes por clase de los conjuntos MNIST y Fashion MNIST.

En general, en los resultados obtenidos se puede notar claramente el efecto que tiene la cantidad de datos sobre el desempeño de cada algoritmo, donde al aumentar el tamaño del subconjunto de entrenamiento los valores de las métricas evaluadas también ven un aumento.

Una de las limitaciones de esta investigación es que las combinaciones que se probaron son solo dos, esto se debe al crecimiento de la cantidad de experimentos necesarios a realizar al aumentar la cantidad de combinaciones.

Aportes, conclusiones y trabajo futuro

Aportes

Los principales aportes de esta tesis son los siguientes:

- Búsqueda e identificación de técnicas utilizadas en la literatura para enfrentar un conjunto pequeño de datos empleando aprendizaje profundo.
- La comparación de las técnicas de aprendizaje profundo con algoritmos de aprendizaje automático clásicos como el árbol de decisión, bosque aleatorio y la máquina de soporte de vectores sobre cinco métricas que permiten obtener una visión general del desempeño de cada algoritmo.
- La ratificación de los resultados obtenidos por las diferentes técnicas presentadas en la literatura con conjuntos pequeños de datos como alternativas viables para enfrentar este problema, tales como el uso de Dropout, Agrupación de promedio global, Transferencia de aprendizaje, Aumento de la cantidad de los datos, Decadencia cíclica de tasa de aprendizaje y la función de costo de Similitud de coseno.
- Se determina que es posible mejorar los resultados obtenidos por algoritmos de aprendizaje profundo cuando se enfrenta un conjunto pequeño de datos superando el desempeño obtenido sin técnicas y por los algoritmos clásicos.
- Se plantea una base para realización trabajos futuros de investigación sobre conjuntos pequeños de datos, agregando más técnicas de aprendizaje profundo o utilizando diferentes

conjuntos de datos.

Conclusiones

El objetivo de esta tesis ha sido el estudio del comportamiento de algoritmos de aprendizaje automático, incluidos algoritmos de aprendizaje clásicos y profundos, cuando se utiliza un conjunto pequeño de datos para su entrenamiento y la búsqueda de una técnica o combinación de técnicas propuestas en la literatura que permitieran la mitigación del problema de la limitación en la cantidad de datos. El motivo fue la dificultad en la obtención de grandes cantidades de datos correctamente etiquetados en algunos campos de la informática, medicina, entre otros.

Se ha realizado una búsqueda en la literatura para encontrar las técnicas de aprendizaje profundo utilizadas para clasificación de imágenes con conjuntos pequeños de imágenes. Luego, se han implementado tres algoritmos clásicos, Árbol de Decisión, Bosque Aleatorio y Máquina de Soporte de Vectores; y nueve algoritmos de Redes Neuronales Profundas. Entre algoritmos clásicos, técnicas de aprendizaje profundo y combinaciones se completa un total de 13 algoritmos diferentes. Estos fueron entrenados sobre cuatro subconjuntos de diferentes tamaños (10, 50, 250 y 500 imágenes por clase) de los conjuntos de datos MNIST, Fashion MNIST y CIFAR-10, los cuales fueron escogidos de manera aleatoria desde el conjunto de entrenamiento de cada conjunto de datos. En total se realizaron 156 experimentos, ya que cada algoritmo se entrena en cada uno de los subconjuntos para cada uno de los tres conjuntos utilizados.

De la experimentación realizada se concluye que es posible encontrar una técnica o combinación que permita mitigar la deficiencia en la cantidad de datos, pero esta técnica o combinación será distinta para cada conjunto de datos, cantidad de datos y, posiblemente, métrica. Por ejemplo, en la Tabla 6.16 se pueden observar las técnicas o algoritmos que obtienen los resultados más altos en las métricas de Exactitud, F1 y MCC. De la tabla se puede notar que existen algunas variaciones de cual es el método más adecuado dependiendo de la métrica que se escoja como más relevante. Por lo tanto, se recomienda que a la hora de buscar la técnica más adecuada, se tenga en cuenta cual es la métrica a optimizar para la tarea dada, de manera que se facilite la elección del algoritmo más adecuado.

		Exactitud		F1		MCC	
	Subconjunto	Más alto	Segundo más alto	Más alto	Segundo más alto	Más alto	Segundo más alto
MNIST	10	EMM	C1	EMM	GAP	EMM	C1
	50	EMM	C1	EMM	C1	EMM	C1
	250	EMM	GAP	EMM	GAP	EMM	GAP
	500	EMM	DNN	EMM	DNN/C2	EMM	DNN
Fashion MNIST	10	EMM	SVM	EMM	SVM	EMM	SVM
	50	EMM	SVM	EMM	SVM	EMM	SVM
	250	EMM	GAP	EMM	GAP	EMM	GAP
	500	EMM	Dropout	EMM	Dropout	EMM	Dropout
CIFAR-10	10	C2	EMM	C2	EMM	C2	EMM
	50	C2	EMM	C2	EMM	C2	EMM
	250	C2	EMM	C2	EMM	C2	EMM
	500	C2	EMM	C2	EMM	C2	EMM

Tabla 6.16: Técnicas y algoritmos con resultados más altos en Exactitud, F1 y MCC para cada subconjunto de datos.

También, de la Tabla 6.16 se observa que las técnicas que obtuvieron mejores resultados en las métricas de Exactitud, F1 y MCC son: Dropout, Agrupación de promedio global, Ensamblaje de múltiples modelos, Combinación de técnicas (C2) que utiliza Transferencia de aprendizaje y Aumento de datos y el algoritmo de Máquina de soporte de vectores. Por lo tanto, se recomienda experimentar con dichas técnicas y algoritmos al enfrentar deficiencia en la cantidad de datos de entrenamiento, ya que se espera que presenten resultados similares a los obtenidos durante la experimentación. Sin embargo, esto último requiere de estudios y experimentación futura para observar si se sostiene en distintos conjuntos pequeños de datos.

Trabajo futuro

Como línea de trabajo futuro se plantea la comparación de una mayor cantidad de permutaciones en las técnicas recopiladas por esta tesis. Otra línea de trabajo futuro que se plantea es el desarrollo de un método automatizado para experimentar con las técnicas de aprendizaje profundo para un conjunto pequeño de datos que permita encontrar las técnicas que se desempeñan mejor para dicho conjunto. Esto implica, tanto la experimentación con la arquitectura de la red como la optimización de los parámetros de la misma.

Referencias

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, y Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv preprint*, 2016. URL http://arxiv.org/abs/1603.04467.

Géron Aurélien. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems.* O'Reilly Media, 2019. ISBN 9781492032649.

Pierre Baldi y Peter Sadowski. Understanding Dropout. Inf. téc., University of California, 2013. URL https://papers.nips.cc/paper/2013/file/71f6278d140af599e06ad9bf1ba03cb0-Paper.pdf.

Björn Barz y Joachim Denzler. Deep Learning is not a Matter of Depth but of Good Training. En *International Conference on Pattern Recognition and Artificial Intelligence (ICPRAI)*. 2018.

Björn Barz y Joachim Denzler. Deep Learning on Small Datasets without Pre-Training using Cosine Loss. *openaccess.thecvf.com*, 2019. URL http://arxiv.org/abs/1901.09054.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. ISSN 08856125. doi: 10.1023/A:1010933404324. URL http://link.springer.com/10.1023/A:1010933404324.

L. Brigato y L. Iocchi. A Close Look at Deep Learning with Small Data. *arXiv preprint*, 2020. URL http://arxiv.org/abs/2003.12843.

Jinchuan Chen, Yueguo Chen, Xiaoyong Du, Cuiping Li, Jiaheng Lu, Suyun Zhao, y Xuan Zhou. Big data challenge: a data management perspective. *Frontiers of Computer Science*, 7(2):157–164, 2013. ISSN 2095-2228. doi:10.1007/s11704-013-3903-7. URL http://link.springer.com/10.1007/s11704-013-3903-7.

Yung-Yao Chen, Yu-Hsiu Lin, Chia-Ching Kung, Ming-Han Chung, y I-Hsuan Yen. Design and Implementation of Cloud Analytics-Assisted Smart Power Meters Considering Advanced Artificial Intelligence as Edge Analytics in Demand-Side Management for Smart Homes. *Sensors*, 19(9):2047, 2019. ISSN 1424-8220. doi:10.3390/s19092047. URL https://www.mdpi.com/1424-8220/19/9/2047.

Davide Chicco y Giuseppe Jurman. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21(1):6, 2020. ISSN 1471-2164. doi:10.1186/s12864-019-6413-7. URL https://link.springer.com/articles/10.1186/s12864-019-6413-7https://link.springer.com/article/10.1186/s12864-019-6413-7https://bmcgenomics.biomedcentral.com/articles/10.1186/s12864-019-6413-7.

Francois Chollet y otros. Keras. 2015. URL https://github.com/keras-team/keras.

Corinna Cortes y Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. ISSN 0885-6125. doi:10.1007/BF00994018. URL https://link.springer.com/article/10.1007/BF00994018http://link.springer.com/10.1007/BF00994018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, y Li Fei-Fei. ImageNet: A large-scale hierarchical image database. En *2009 IEEE Conference on Computer Vision and Pattern*

Recognition, págs. 248–255. IEEE, 2009. ISBN 978-1-4244-3992-8. doi:10.1109/CVPR.2009.5206848. URL https://ieeexplore.ieee.org/document/5206848/.

Rhett N. D'souza, Po-Yao Huang, y Fang-Cheng Yeh. Structural Analysis and Optimization of Convolutional Neural Networks with a Small Sample Size. *Scientific Reports*, 10(1):834, 2020. ISSN 2045-2322. doi:10.1038/s41598-020-57866-2. URL http://www.nature.com/articles/s41598-020-57866-2.

Vincent Dumoulin y Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint*, 2016. URL http://arxiv.org/abs/1603.07285.

Shuo Feng, Huiyu Zhou, y Hongbiao Dong. Using deep neural network with small dataset to predict material defects. *Materials & Design*, 162:300–310, 2019. ISSN 02641275. doi:10.1016/j.matdes.2018.11.060. URL https://www.sciencedirect.com/science/article/pii/S0264127518308682.

Raúl Gómez. Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and all those confusing names. 2018. URL https://gombru.github.io/2018/05/23/cross_entropy_loss/.

Ian Goodfellow, Yoshua Bengio, y Aaron Courville. *Deep learning*. MIT Press, 2016. ISBN 0262035618.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, y Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. ISSN 0001-0782. doi:10.1145/3422622. URL https://dl.acm.org/doi/10.1145/3422622.

Ophir Gozes y Hayit Greenspan. Deep Feature Learning from a Hospital-Scale Chest X-ray Dataset with Application to TB Detection on a Small-Scale Dataset. En *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, págs. 4076–4079. IEEE, 2019. ISBN 978-1-5386-1311-5. ISSN 1557170X. doi:10.1109/EMBC.2019.8856729. URL https://ieeexplore.ieee.org/document/8856729/.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, y Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint*, 2012. URL http://arxiv.org/abs/1207.0580.

Mohammad Hossin y Nasir Sulaiman. A Review on Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process*, 5(2):01–11, 2015. ISSN 2231007X. doi:10.5121/ijdkp.2015.5201. URL http://www.aircconline.com/ijdkp/V5N2/5215ijdkp01.pdf.

Sergey Ioffe y Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *32nd International Conference on Machine Learning, ICML 2015*, 1:448–456, 2015. URL http://arxiv.org/abs/1502.03167.

Vijay Kotu y Bala Deshpande. Data Mining Process. En *Predictive Analytics and Data Mining*, págs. 17–36. Elsevier, 2015. doi:10.1016/B978-0-12-801460-8.00002-1. URL https://linkinghub.elsevier.com/retrieve/pii/B9780128014608000021.

Xinyi Le, Junhui Mei, Haodong Zhang, Boyu Zhou, y Juntong Xi. A learning-based approach for surface defect detection using small image datasets. *Neurocomputing*, 2020. ISSN 09252312. doi:10.1016/j.neucom.2019.09.107. URL https://linkinghub.elsevier.com/retrieve/pii/S0925231220303386.

Hyunkwang Lee, Sehyo Yune, Mohammad Mansouri, Myeongchan Kim, Shahein H. Tajmir, Claude E. Guerrier, Sarah A. Ebert, Stuart R. Pomerantz, Javier M. Romero, Shahmir Kamalian, Ramon G. Gonzalez, Michael H. Lev, y Synho Do. An explainable deep-learning algorithm for the detection of acute intracranial haemorrhage from small datasets. *Nature Biomedical Engineering*, 3(3):173–182, 2019. ISSN 2157-846X. doi:10.1038/s41551-018-0324-9. URL http://www.nature.com/articles/s41551-018-0324-9.

Min Lin, Qiang Chen, y Shuicheng Yan. Network In Network. *arXiv preprint*, pág. 10, 2013. URL http://arxiv.org/abs/1312.4400.

Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen A.W.M. van der Laak, Bram van Ginneken, y Clara I. Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42:60–88, 2017. ISSN 13618415. doi:10.1016/j.media.2017.07.005. URL https://linkinghub.elsevier.com/retrieve/pii/S1361841517301135.

Changchong Lu y Weihai Li. Ship Classification in High-Resolution SAR Images via Transfer Learning with Small Training Dataset. *Sensors*, 19(1):63, 2018. ISSN 1424-8220. doi:10.3390/s19010063. URL http://www.mdpi.com/1424-8220/19/1/63.

B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *BBA - Protein Structure*, 405(2):442–451, 1975. ISSN 00052795. doi:10.1016/0005-2795(75)90109-9. URL https://linkinghub.elsevier.com/retrieve/pii/0005279575901099.

Tom M Mitchell. *Machine Learning*. McGraw-Hill, 1997. ISBN 0070428077.

Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, y Stephen Marshall. Activation Functions: Comparison of trends in Practice and Research for Deep Learning. *arXiv preprint*, 2018. URL http://arxiv.org/abs/1811.03378.

Felipe Oviedo, Zekun Ren, Shijing Sun, Charles Settens, Zhe Liu, Noor Titan Putri Hartono, Savitha Ramasamy, Brian L. DeCost, Siyu I. P. Tian, Giuseppe Romano, Aaron Gilad Kusne, y Tonio Buonassisi. Fast and interpretable classification of small X-ray diffraction datasets using data augmentation and deep neural networks. *npj Computational Materials*, 5(1):60, 2019. ISSN 2057-3960. doi:10.1038/s41524-019-0196-x. URL http://www.nature.com/articles/s41524-019-0196-x.

Kitsuchart Pasupa y Wisuwat Sunhem. A comparison between shallow and deep architecture classifiers on small dataset. En *2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE)*, págs. 1–6. IEEE, 2016. ISBN 978-1-5090-4139-8. doi:10.1109/ICITEED.2016.7863293. URL http://ieeexplore.ieee.org/document/7863293/.

F Pedregosa, G Varoquaux, A Gramfort, V Michel, B Thirion, O Grisel, M Blondel, P Prettenhofer, R Weiss, V Dubourg, J Vanderplas, A Passos, D Cournapeau, M Brucher, M Perrot, y E Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Peltarion. 2D Global average pooling. 2019. URL https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/blocks/2d-global-average-pooling.

Gil Press. 7 Indicators Of The State-Of-Artificial Intelligence (AI), April 2019. 2019. URL https://www.forbes.com/sites/gilpress/2019/04/30/7-indicators-of-the-state-of-artificial-intelligence-ai-april-2019/#10e0e2926009.

Pranav Rajpurkar, Allison Park, Jeremy Irvin, Chris Chute, Michael Bereket, Domenico Mastrodicasa, Curtis P. Langlotz, Matthew P. Lungren, Andrew Y. Ng, y Bhavik N. Patel. AppendiXNet: Deep Learning for Diagnosis of Appendicitis from A Small Dataset of CT Exams Using Video Pretraining. *Scientific Reports*, 10(1):3958, 2020. ISSN 2045-2322. doi:10.1038/s41598-020-61055-6. URL http://www.nature.com/articles/s41598-020-61055-6.

Anh H. Reynolds. Convolutional Neural Networks (CNNs). 2019. URL https://anhreynolds.com/blogs/cnn.html.

Lior Rokach y Oded Maimon. Decision Trees. En *Data Mining and Knowledge Discovery Handbook*, págs. 165–192. Springer-Verlag, New York, 2006. doi:10.1007/0-387-25465-X_9. URL http://link.springer.com/10.1007/0-387-25465-X_9.

Lorenzo Rosasco, Ernesto De Vito, Andrea Caponnetto, Michele Piana, y Alessandro Verri. Are Loss Functions All the Same? *Neural Computation*, 16(5):1063–1076, 2004. ISSN 0899-7667. doi:10.1162/089976604773135104. URL https://www.mitpressjournals.org/doi/abs/10.1162/089976604773135104.

F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958. ISSN 1939-1471. doi:10.1037/h0042519. URL http://doi.apa.org/getdoi.cfm?doi=10.1037/h0042519.

David E. Rumelhart, Geoffrey E. Hinton, y Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. ISSN 0028-0836. doi:10.1038/323533a0. URL http://www.nature.com/articles/323533a0.

Leslie N. Smith. Cyclical Learning Rates for Training Neural Networks. *Proceedings - 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017*, págs. 464–472, 2015. URL http://arxiv.org/abs/1506.01186.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, y Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014. URL https://jmlr.org/papers/v15/srivastava14a.html.

Mingxing Tan y Quoc V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *36th International Conference on Machine Learning, ICML 2019*, 2019-June:10691–10700, 2019. URL http://arxiv.org/abs/1905.11946.

C.Z. Tang y H.K. Kwan. Multilayer feedforward neural networks with single powers-of-two weights. *IEEE Transactions on Signal Processing*, 41(8):2724–2727, 1993. ISSN 1053587X. doi:10.1109/78.229903. URL http://ieeexplore.ieee.org/document/229903/.

Thomas Wood. Softmax Function. s.f. URL https://deepai.org/machine-learning-glossary-and-terms/softmax-layer.

Jie Yang, Yuchen Xie, Lin Liu, Bin Xia, Zhanqiang Cao, y Chuanbin Guo. Automated Dental Image Analysis by Deep Learning on Small Dataset. En *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, tomo 1, págs. 492–497. IEEE, 2018. ISBN 978-1-5386-2666-5. ISSN 07303157. doi:10.1109/COMPSAC.2018.00076. URL https://ieeexplore.ieee.org/document/8377701/.

Zijiang Yang, Reda Al-Bahrani, Andrew C. E. Reid, Stefanos Papanikolaou, Surya R. Kalidindi, Wei-keng Liao, Alok Choudhary, y Ankit Agrawal. Deep learning based domain knowledge integration for small datasets: Illustrative applications in materials informatics. En *2019 International Joint Conference on Neural Networks (IJCNN)*, tomo 2019-July, págs. 1–8. IEEE, 2019. ISBN 978-1-7281-1985-4. doi:10.1109/IJCNN.2019.8852162. URL https://ieeexplore.ieee.org/document/8852162/.

Fisher Yu y Vladlen Koltun. Multi-Scale Context Aggregation by Dilated Convolutions. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1511.07122.

Shu Zhang, Fangfang Han, Zhengrong Liang, Jiaxing Tan, Weiguo Cao, Yongfeng Gao, Marc Pomeroy, Kenneth Ng, y Wei Hou. An investigation of CNN models for differentiating malignant from benign lesions using small pathologically proven datasets. *Computerized Medical Imaging and Graphics*, 77:101645, 2019. ISSN 08956111. doi:10.1016/j.compmedimag.2019.101645. URL https://linkinghub.elsevier.com/retrieve/pii/S0895611119300643.

Shengwei Zhou, Caikou Chen, Guojiang Han, y Xielian Hou. Deep Convolutional Neural Network with Dilated Convolution Using Small Size Dataset. En *2019 Chinese Control Conference (CCC)*, tomo 2019-July, págs. 8568–8572. IEEE, 2019. ISBN 978-9-8815-6397-2. ISSN 21612927. doi:10.23919/ChiCC.2019.8865226. URL https://ieeexplore.ieee.org/document/8865226/.

Apéndice A

Configuración de los algoritmos

En este Apéndice se presenta la configuración de cada algoritmo entrenado para cada conjunto de datos utilizado y, también, se agregan los resultados obtenidos para dicha configuración. La configuración se presenta de la siguiente manera: para los algoritmos clásicos, se presentan dos tablas, la primera, muestra los parámetros utilizados en su implementación con Scikit-learn y la segunda tabla muestra los resultados obtenidos en las métricas; para los algoritmos profundos se presentan tres tablas, la primera, con los parámetros que se utilizan para la implementación con Tensorflow y Keras, la segunda, con la arquitectura de la red neuronal utilizada y, por último, la tercera tabla presenta los resultados obtenidos en las diferentes métricas mencionadas previamente. Además se añade el gráfico de costo/perdida por iteración durante el entrenamiento.

A.1. Configuración de algoritmos entrenados sobre subconjuntos de MNIST

A.1.1. Árbol de decisión

Los parámetros utilizados para la implementación del Árbol de decisión se presentan en la Tabla A.1.

	Ejemplos por clase			
Parámetros	10	50	250	500
criterion	entropy	entropy	entropy	entropy
max_depth	8	12	12	12
min_samples_leaf	2	1	3	5
min_samples_split	2	4	5	4
random_state	0	0	0	0

Tabla A.1: Tabla de parámetros para Árbol de decisión sobre MNIST.

Los resultados obtenidos por el Árbol de decisión se presentan en la Tabla A.2.

Ejemplos/Clase: 10	
Exactitud	0.418
Precisión	0.425
Recall	0.412
F1	0.409
MCC	0.355

Ejemplos/Clase: 50	
Exactitud	0.601
Precisión	0.602
Recall	0.597
F1	0.596
MCC	0.557

Ejemplos/Clase: 250	
Exactitud	0.740
Precisión	0.737
Recall	0.737
F1	0.736
MCC	0.711

Ejemplos/Clase: 500	
Exactitud	0.789
Precisión	0.786
Recall	0.787
F1	0.786
MCC	0.766

Tabla A.2: Tabla de resultados de Árbol de decisión sobre MNIST.

A.1.2. Bosque Aleatorio

Los parámetros utilizados para la implementación del Bosque aleatorio se presentan en la Tabla A.3.

	Ejemplos por clase			
Parámetros	10	50	250	500
criterion	entropy	gini	gini	gini
min_samples_leaf	1	1	1	1
min_samples_split	6	2	2	2
n_estimators	500	500	2500	5000
random_state	0	0	0	0

Tabla A.3: Tabla de parámetros para Bosque Aleatorio sobre MNIST.

Los resultados obtenidos por el algoritmo Bosque aleatorio se presentan en la Tabla A.4.

Ejemplos/Clase: 10	
Exactitud	0.749
Precisión	0.748
Recall	0.745
F1	0.741
MCC	0.722

Ejemplos/Clase: 50	
Exactitud	0.890
Precisión	0.889
Recall	0.888
F1	0.888
MCC	0.878

Ejemplos/Clase: 250	
Exactitud	0.934
Precisión	0.934
Recall	0.934
F1	0.933
MCC	0.927

Ejemplos/Clase: 500	
Exactitud	0.946
Precisión	0.945
Recall	0.945
F1	0.945
MCC	0.94

Tabla A.4: Tabla de resultados de Bosque Aleatorio sobre MNIST.

A.1.3. Máquina de Soporte de Vectores

Los parámetros utilizados para la implementación de la Máquina de Soporte de Vectores se presentan en la Tabla A.5.

	Ejemplos por clase			
Parámetros	10	50	250	500
C	2	5	2	5
degree	1	1	1	1
kernel	rbf	rbf	rbf	rbf
random_state	0	0	0	0

Tabla A.5: Tabla de parámetros para Máquina de Soporte de Vectores sobre MNIST.

Los resultados obtenidos por la Máquina de Soporte de Vectores se presentan en la Tabla A.4.

Ejemplos/Clase: 10	
Exactitud	0.792
Precisión	0.792
Recall	0.788
F1	0.787
MCC	0.769

Ejemplos/Clase: 50	
Exactitud	0.903
Precisión	0.902
Recall	0.902
F1	0.902
MCC	0.892

Ejemplos/Clase: 250	
Exactitud	0.946
Precisión	0.946
Recall	0.946
F1	0.946
MCC	0.94

Ejemplos/Clase: 500	
Exactitud	0.960
Precisión	0.959
Recall	0.959
F1	0.959
MCC	0.955

Tabla A.6: Tabla de resultados de Máquina de Soporte de Vectores sobre MNIST.

A.1.4. Red Neuronal Profunda

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.7.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	10	8	7	7
Tamaño Lote	32			
Tasa de Aprendizaje	0.001			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.7: Tabla de parámetros para Red Neuronal Profunda sobre MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.8.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	28x28x1				
Convolucional		(7,7)	64		ReLU
AgrupaciónMáxima		(4,4)			
Convolucional		(5,5)	128		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(7,7)	32		ReLU
Convolucional		(5,5)	128		ReLU
Aplanar					
Densa				32	ReLU
Densa				32	ReLU
Densa (Salida)				10	Softmax

Tabla A.8: Tabla resumen de la Estructura de la Red Neuronal Profunda sobre MNIST.

Los resultados obtenidos por la Red Neuronal Profunda se presentan en la Tabla A.9.

Ejemplos/Clase: 10	
Exactitud	0.680
Precisión	0.677
Recall	0.671
F1	0.666
MCC	0.646

Ejemplos/Clase: 50	
Exactitud	0.920
Precisión	0.921
Recall	0.919
F1	0.919
MCC	0.911

Ejemplos/Clase: 250	
Exactitud	0.968
Precisión	0.968
Recall	0.968
F1	0.968
MCC	0.964

Ejemplos/Clase: 500	
Exactitud	0.982
Precisión	0.982
Recall	0.981
F1	0.981
MCC	0.98

Tabla A.9: Resultados de la Red Neuronal Profunda sobre MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.1.

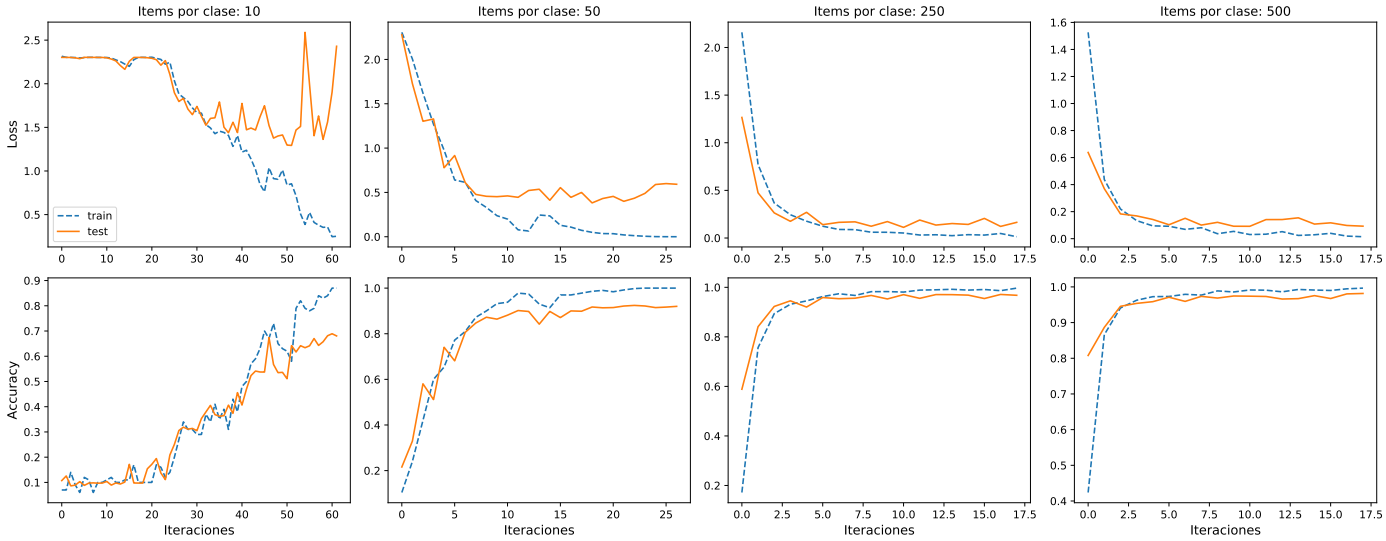

Figura A.1: Gráficas de costo (Loss) y Exactitud (Accuracy) durante entrenamiento para Red Neuronal Profunda sobre MNIST.

A.1.5. Red Neuronal Profunda con Dropout

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.10.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	15	10	10	5
Tamaño Lote	20			32
Tasa de Aprendizaje	0.001			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.10: Tabla de parámetros para Red Neuronal Profunda con Dropout sobre MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.11.

Capa	Tamaño	Kernel	Filtros	Probabilidad	Unidades	Activación
Entrada	28x28x1					
Convolucional		(7,7)	64			ReLU
AgrupaciónMáxima		(4,4)				
Convolucional		(5,5)	128			ReLU
Convolucional		(5,5)	64			ReLU
Convolucional		(5,5)	64			ReLU
Convolucional		(7,7)	32			ReLU
Convolucional		(5,5)	128			ReLU
Aplanar						
Dropout				0.25		
Densa					128	
Dropout				0.5		
Densa					64	
Densa					10	Softmax

Tabla A.11: Tabla resumen de la Estructura de Red Neuronal Profunda con Dropout sobre MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con Dropout se presentan en la Tabla A.12.

Ejemplos/Clase: 10	
Exactitud	0.746
Precisión	0.775
Recall	0.741
F1	0.737
MCC	0.722

Ejemplos/Clase: 50	
Exactitud	0.903
Precisión	0.916
Recall	0.901
F1	0.903
MCC	0.894

Ejemplos/Clase: 250	
Exactitud	0.974
Precisión	0.974
Recall	0.974
F1	0.974
MCC	0.971

Ejemplos/Clase: 500	
Exactitud	0.979
Precisión	0.98
Recall	0.979
F1	0.979
MCC	0.977

Tabla A.12: Resultados de la Red Neuronal Profunda con Dropout sobre MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.2.

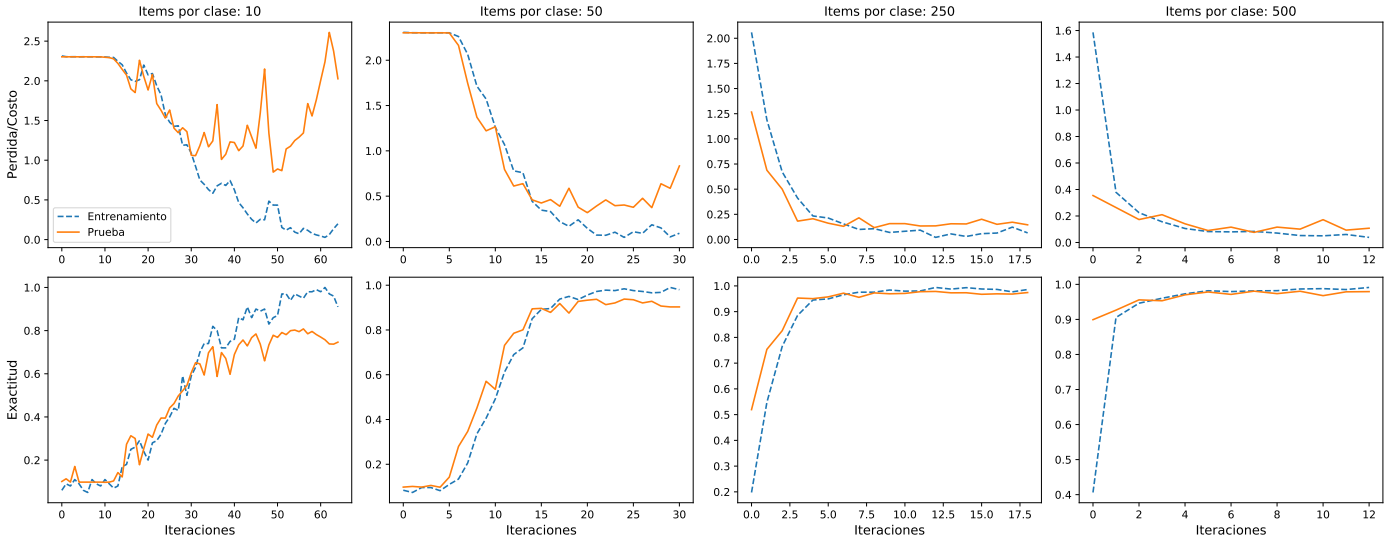

Figura A.2: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Dropout sobre MNIST.

A.1.6. Red Neuronal Profunda con capa Agrupación Promedio Global

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.13.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	15	10	15	10
Tamaño Lote	20	32	64	
Tasa de Aprendizaje	0.001			
Limite de iteraciones (Entrenamiento)	60			

Tabla A.13: Tabla de parámetros para Red Neuronal Profunda con Capa Agrupación Promedio Global sobre MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.14.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	28x28x1				
Convolucional		(7,7)	128		ReLU
AgrupaciónMáxima		(4,4)			
Convolucional		(5,5)	128		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(7,7)	32		ReLU
Convolucional		(5,5)	128		ReLU
AgrupaciónPromedioGlobal					
Densa				10	Softmax

Tabla A.14: Tabla resumen de la Estructura de Red Neuronal Profunda con Capa Agrupación Promedio Global sobre MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con capa de Agrupación Promedio Global se presentan en la Tabla A.15.

Ejemplos/Clase: 10	
Exactitud	0.803
Precisión	0.813
Recall	0.801
F1	0.802
MCC	0.783

Ejemplos/Clase: 50	
Exactitud	0.902
Precisión	0.916
Recall	0.897
F1	0.898
MCC	0.892

Ejemplos/Clase: 250	
Exactitud	0.982
Precisión	0.982
Recall	0.982
F1	0.982
MCC	0.98

Ejemplos/Clase: 500	
Exactitud	0.980
Precisión	0.98
Recall	0.98
F1	0.980
MCC	0.978

Tabla A.15: Resultados de la Red Neuronal Profunda con Capa Agrupación Promedio Global sobre MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.3.

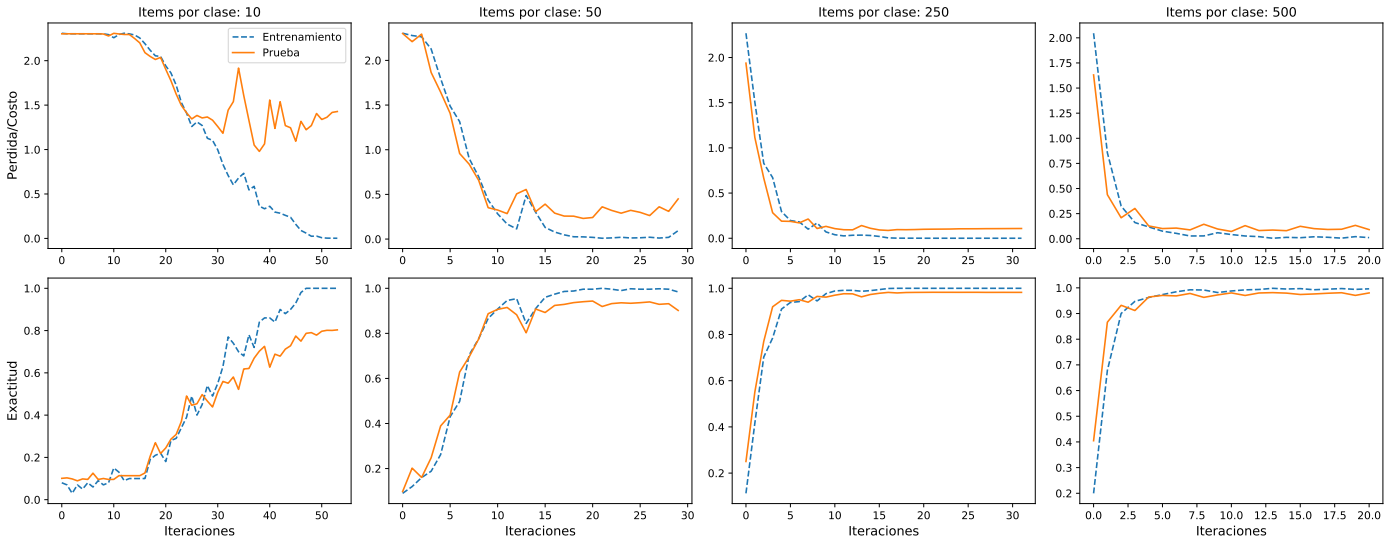

Figura A.3: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Capa Agrupación Promedio Global sobre MNIST.

A.1.7. Red Neuronal Profunda con Normalización por lotes

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.16.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	10	10	7	7
Tamaño Lote	16	24	32	
Tasa de Aprendizaje	0.004			
Limite de iteraciones (Entrenamiento)	60			

Tabla A.16: Tabla de parámetros para Red Neuronal Profunda con Normalización por Lotes sobre MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.17.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	28x28x1				
Convolucional		(7,7)	64		ReLU
NormalizaciónPorLotes					
AgrupaciónMáxima		(4,4)			
NormalizaciónPorLotes					
Convolucional		(5,5)	128		ReLU
NormalizaciónPorLotes					
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(7,7)	32		ReLU
Convolucional		(5,5)	128		ReLU
Aplanar					
Densa				32	ReLU
Densa				32	ReLU
Densa				10	Softmax

Tabla A.17: Tabla resumen de la Estructura de Red Neuronal Profunda con Normalización por Lotes sobre MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con capa de Normalización por Lotes se presentan en la Tabla A.18.

Ejemplos/Clase: 10	
Exactitud	0.674
Precisión	0.712
Recall	0.671
F1	0.676
MCC	0.642

Ejemplos/Clase: 50	
Exactitud	0.914
Precisión	0.916
Recall	0.912
F1	0.913
MCC	0.904

Ejemplos/Clase: 250	
Exactitud	0.940
Precisión	0.943
Recall	0.939
F1	0.939
MCC	0.934

Ejemplos/Clase: 500	
Exactitud	0.954
Precisión	0.956
Recall	0.955
F1	0.955
MCC	0.949

Tabla A.18: Resultados de la Red Neuronal Profunda con Normalización por Lotes sobre MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.4.

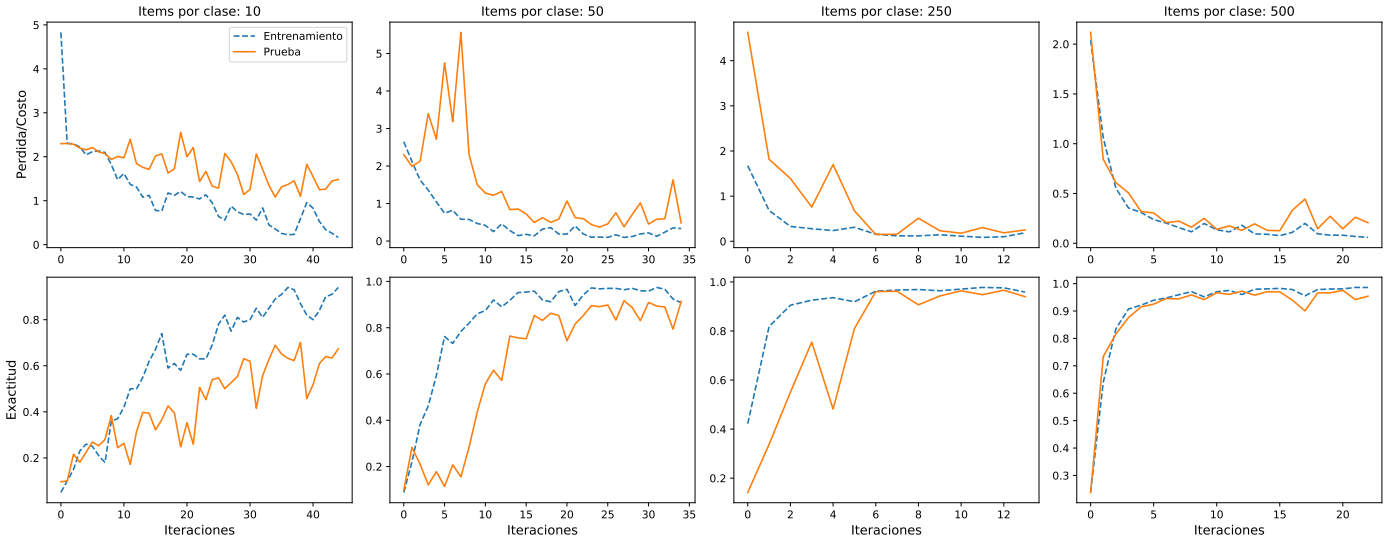

Figura A.4: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Normalización por Lotes sobre MNIST.

A.1.8. Red Neuronal Profunda con función de costo Similitud de Coseno

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.19.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	10	8	6	4
Tamaño Lote	32			
Tasa de Aprendizaje	0.001			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.19: Tabla de parámetros para Red Neuronal Profunda con función de costo Similitud de Coseno sobre MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.20.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	28x28x1				
Convolucional		(7,7)	64		ReLU
AgrupaciónMáxima		(4,4)			
Convolucional		(5,5)	128		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(7,7)	32		ReLU
Convolucional		(5,5)	128		ReLU
Aplanar					
Densa				32	ReLU
Densa				32	ReLU
Densa (Salida)				10	Softmax

Tabla A.20: Tabla resumen de la Estructura de Red Neuronal Profunda con función de costo Similitud de Coseno sobre MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con función de costo Similitud de Coseno se presentan en la Tabla A.21.

Ejemplos/Clase: 10	
Exactitud	0.703
Precisión	0.715
Recall	0.698
F1	0.699
MCC	0.671

Ejemplos/Clase: 50	
Exactitud	0.894
Precisión	0.898
Recall	0.892
F1	0.893
MCC	0.883

Ejemplos/Clase: 250	
Exactitud	0.961
Precisión	0.962
Recall	0.961
F1	0.961
MCC	0.957

Ejemplos/Clase: 500	
Exactitud	0.961
Precisión	0.962
Recall	0.961
F1	0.961
MCC	0.957

Tabla A.21: Resultados de la Red Neuronal Profunda con función de costo Similitud de Coseno sobre MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.5.

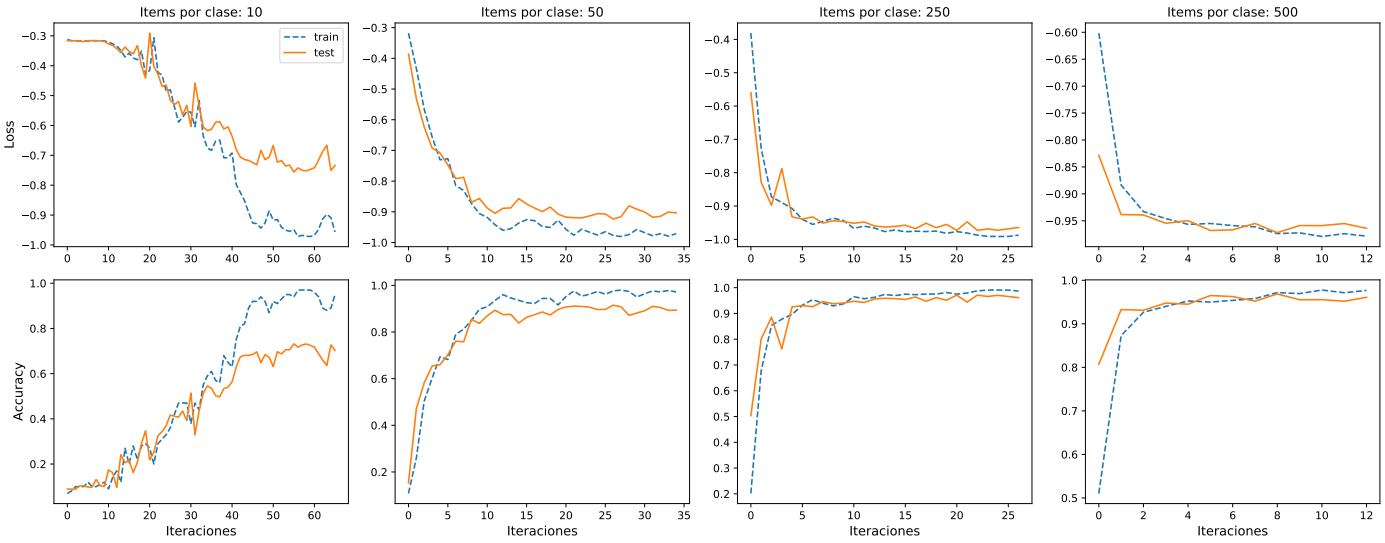

Figura A.5: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con función de costo Similitud de Coseno sobre MNIST.

A.1.9. Red Neuronal Profunda con Convolución Dilatada

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.22.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	5	8	6	5
Tamaño Lote	32			
Tasa de Aprendizaje	0.001			
Limite de iteraciones (Entrenamiento)	60			

Tabla A.22: Tabla de parámetros para Red Neuronal Profunda con Convolución Dilatada sobre MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.23.

Capa	Tamaño	Kernel	Filtros	Tasa de Dilatación	Unidades	Activación
Entrada	28x28x1					
Convolucional		(7,7)	64	2		ReLU
AgrupaciónMáxima		(4,4)				
Convolucional		(5,5)	128	2		ReLU
Convolucional		(5,5)	64	2		ReLU
Convolucional		(5,5)	64	2		ReLU
Convolucional		(7,7)	32	2		ReLU
Convolucional		(5,5)	128	2		ReLU
Aplanar						
Densa					32	ReLU
Densa					32	ReLU
Densa (Salida)					10	Softmax

Tabla A.23: Tabla resumen de la Estructura de Red Neuronal Profunda con Convolución Dilatada sobre MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con Convolución Dilatada se presentan en la Tabla A.24.

Ejemplos/Clase: 10	
Exactitud	0.693
Precisión	0.693
Recall	0.686
F1	0.683
MCC	0.661

Ejemplos/Clase: 50	
Exactitud	0.883
Precisión	0.882
Recall	0.88
F1	0.881
MCC	0.87

Ejemplos/Clase: 250	
Exactitud	0.957
Precisión	0.957
Recall	0.956
F1	0.957
MCC	0.952

Ejemplos/Clase: 500	
Exactitud	0.972
Precisión	0.972
Recall	0.972
F1	0.972
MCC	0.969

Tabla A.24: Resultados de la Red Neuronal Profunda con Convolución Dilatada sobre MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.6.

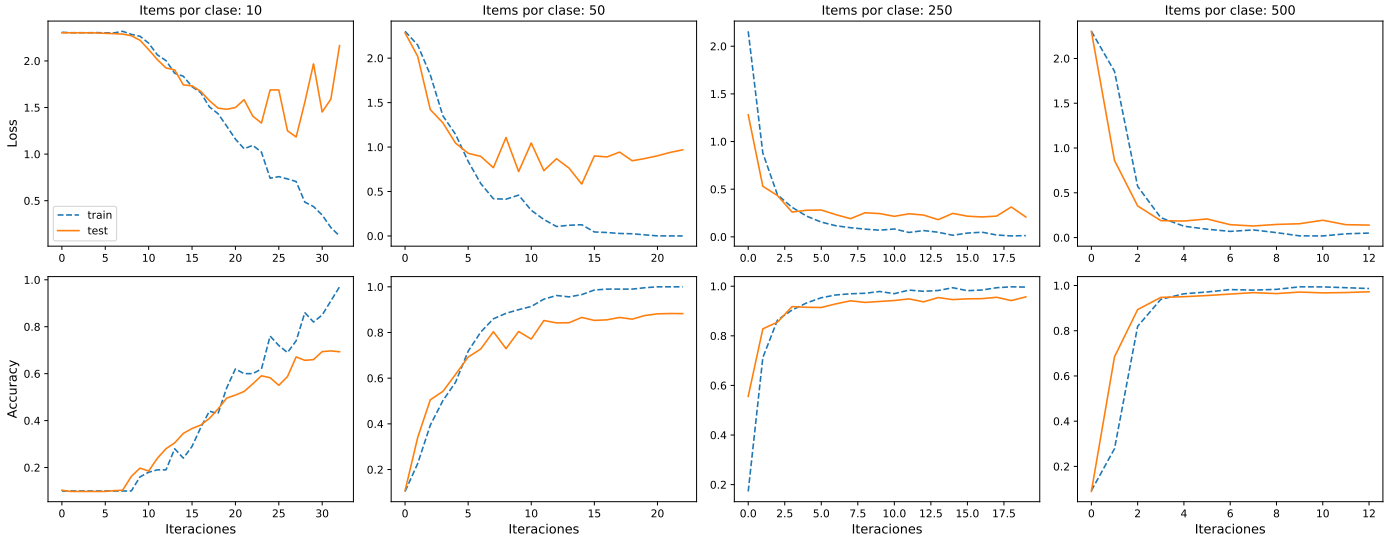

Figura A.6: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Convolución Dilatada sobre MNIST.

A.1.10. Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.25.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	10	8	6	5
Tamaño Lote	32			
Tasa de Aprendizaje mínima	0.00001			
Tasa de Aprendizaje máxima	0.001			
Delta (Decadencia cíclica de tasa de aprendizaje)	10			
s (Decadencia cíclica de tasa de aprendizaje)	10			
Limite de iteraciones (Entrenamiento)	50			

Tabla A.25: Tabla de parámetros para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.26.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	28x28x1				
Convolucional		(7,7)	64		ReLU
AgrupaciónMáxima		(4,4)			
Convolucional		(5,5)	128		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(7,7)	32		ReLU
Convolucional		(5,5)	128		ReLU
Aplanar					
Densa				32	ReLU
Densa				32	ReLU
Densa (Salida)				10	Softmax

Tabla A.26: Tabla resumen de la Estructura de Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje se presentan en la Tabla A.27.

Ejemplos/Clase: 10	
Exactitud	0.692
Precisión	0.701
Recall	0.687
F1	0.683
MCC	0.661

Ejemplos/Clase: 50	
Exactitud	0.890
Precisión	0.89
Recall	0.889
F1	0.889
MCC	0.878

Ejemplos/Clase: 250	
Exactitud	0.967
Precisión	0.967
Recall	0.967
F1	0.967
MCC	0.964

Ejemplos/Clase: 500	
Exactitud	0.978
Precisión	0.978
Recall	0.978
F1	0.978
MCC	0.976

Tabla A.27: Resultados de la Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.7.

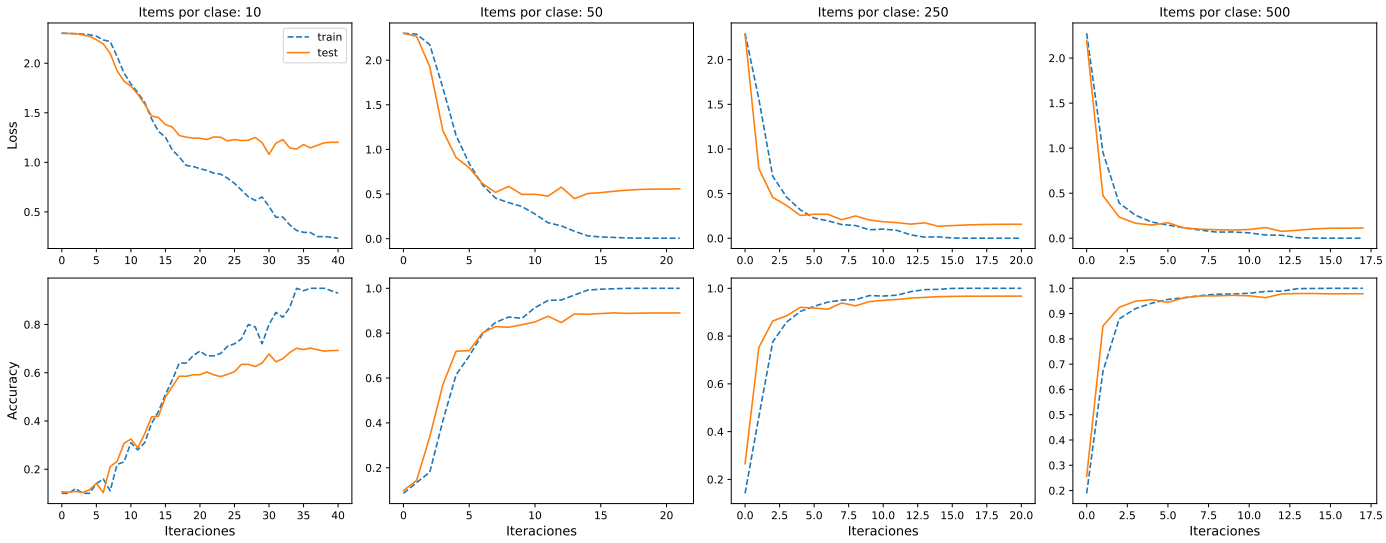

Figura A.7: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre MNIST.

A.1.11. Red Neuronal Profunda con combinación de técnicas (C1)

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.28.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	40	20	15	15
Tamaño Lote	20	48	128	128
Tasa de Aprendizaje mínima	0.0001			
Tasa de Aprendizaje máxima	0.0035			
Delta (Decadencia cíclica de tasa de aprendizaje)	10			
s (Decadencia cíclica de tasa de aprendizaje)	10			
Limite de iteraciones (Entrenamiento)	350			

Tabla A.28: Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas sobre MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.29.

Capa	Tamaño	Kernel	Filtros	Tasa de Dilatación	Probabilidad	Unidades	Activación
Entrada	28x28x1						
Convolucional		(7,7)	64	2			ReLU
NormalizaciónPorLotes							
AgrupaciónMáxima		(4,4)					
Convolucional		(5,5)	128	2			ReLU
Convolucional		(5,5)	64	2			ReLU
Convolucional		(5,5)	64	2			ReLU
Convolucional		(7,7)	32	2			ReLU
Convolucional		(5,5)	128	2			ReLU
AgrupaciónPromedioGlobal							
Dropout					0.1		
Densa						10	Softmax

Tabla A.29: Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas sobre MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con combinación de técnicas se presentan en la Tabla A.30.

Ejemplos/Clase: 10	
Exactitud	0.804
Precisión	0.811
Recall	0.799
F1	0.798
MCC	0.783

Ejemplos/Clase: 50	
Exactitud	0.938
Precisión	0.939
Recall	0.937
F1	0.937
MCC	0.931

Ejemplos/Clase: 250	
Exactitud	0.973
Precisión	0.973
Recall	0.973
F1	0.973
MCC	0.97

Ejemplos/Clase: 500	
Exactitud	0.976
Precisión	0.976
Recall	0.976
F1	0.976
MCC	0.973

Tabla A.30: Resultados de la Red Neuronal Profunda con combinación de técnicas sobre MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.8.

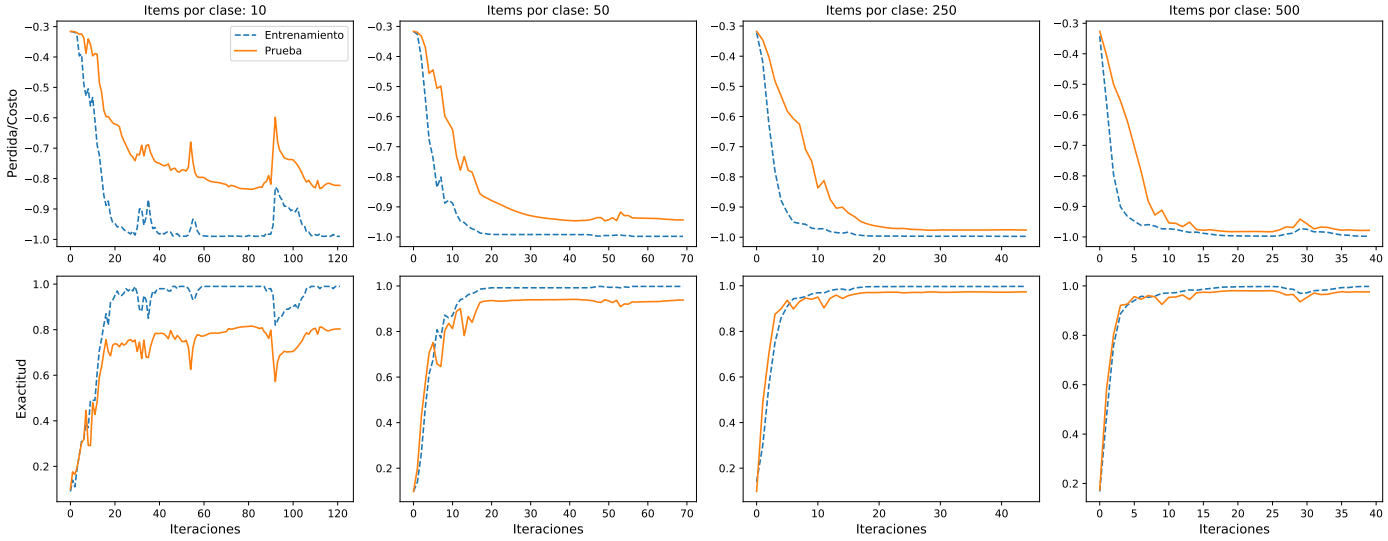

Figura A.8: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas sobre MNIST.

A.1.12. Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos (C2)

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.31.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	15	15	15	10
Tamaño Lote	24	32		
Interpolación	Bilineal			
Rango de rotación (Aumento de datos)	0º a 25º			
Desplazamiento horizontal (Aumento de datos)	10 %			
Desplazamiento vertical (Aumento de datos)	10 %			
Rango de acercamiento (Aumento de datos)	85 % a 100 %			
Volteo horizontal (Aumento de datos)	No			
Tasa de Aprendizaje mínima	0.0001			
Tasa de Aprendizaje máxima	0.001			
Delta (Decadencia cíclica de tasa de aprendizaje)	10			
s (Decadencia cíclica de tasa de aprendizaje)	10			
Limite de iteraciones (Entrenamiento)	100			

Tabla A.31: Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.32.

Capa	Tamaño	Tasa de Aumento	Probabilidad	Unidades	Activación
Entrada	28x28x3				
Ampliar		(8,8)			
EfficientNetB0					
AgrupaciónPromedioGlobal					
NormalizaciónPorLotes					
Dropout			0.5		
Densa				10	Softmax

Tabla A.32: Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con combinación de técnicas más

Transferencia de aprendizaje y Aumento artificial de datos se presentan en la Tabla A.33.

Ejemplos/Clase: 10	
Exactitud	0.796
Precisión	0.811
Recall	0.791
F1	0.791
MCC	0.775

Ejemplos/Clase: 50	
Exactitud	0.930
Precisión	0.93
Recall	0.929
F1	0.929
MCC	0.922

Ejemplos/Clase: 250	
Exactitud	0.967
Precisión	0.967
Recall	0.967
F1	0.967
MCC	0.963

Ejemplos/Clase: 500	
Exactitud	0.981
Precisión	0.981
Recall	0.981
F1	0.981
MCC	0.979

Tabla A.33: Resultados de la Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.9.

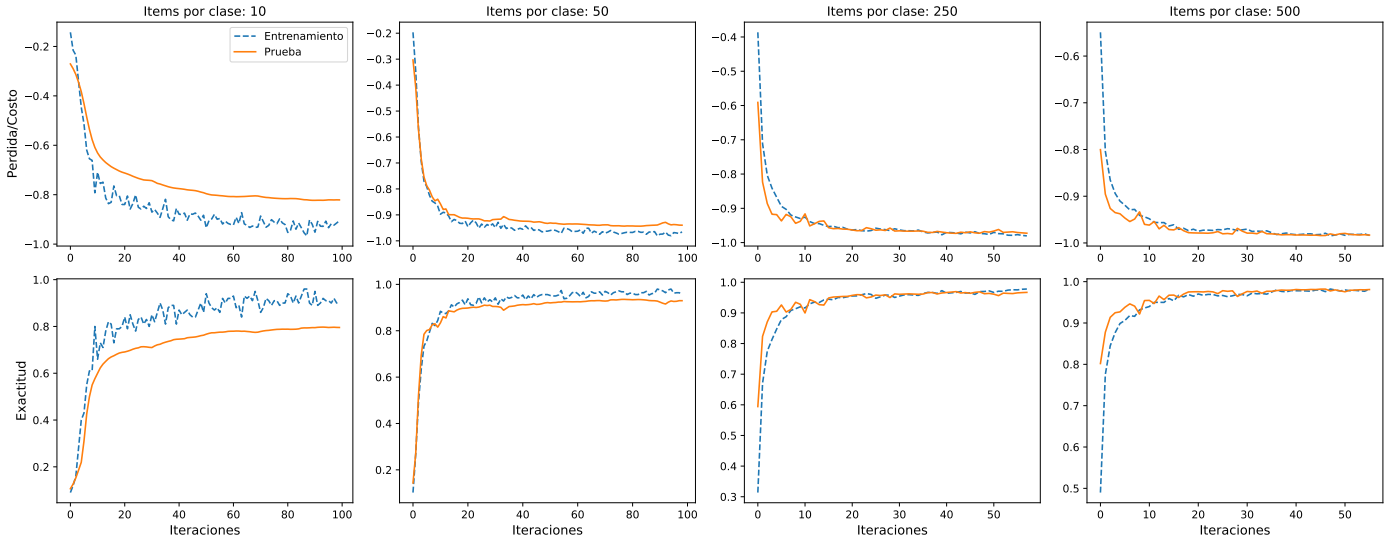

Figura A.9: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre MNIST.

A.1.13. Ensamblaje de todas las Redes Neuronales Profundas

Los resultados obtenidos por el ensamblaje de redes neuronales se presentan en la Tabla A.34.

Ejemplos/Clase: 10	
Exactitud	0.868
Precisión	0.875
Recall	0.863
F1	0.862
MCC	0.854

Ejemplos/Clase: 50	
Exactitud	0.960
Precisión	0.960
Recall	0.959
F1	0.959
MCC	0.955

Ejemplos/Clase: 250	
Exactitud	0.986
Precisión	0.986
Recall	0.986
F1	0.986
MCC	0.985

Ejemplos/Clase: 500	
Exactitud	0.990
Precisión	0.990
Recall	0.990
F1	0.990
MCC	0.989

Tabla A.34: Resultados del ensamblaje de todas las Redes Neuronales Profundas sobre MNIST.

A.2. Configuración de algoritmos entrenados sobre subconjuntos de Fashion MNIST

A.2.1. Árbol de decisión

Los parámetros utilizados para la implementación del Árbol de decisión se presentan en la Tabla A.35.

	Ejemplos por clase			
Parámetros	10	50	250	500
criterion	gini	entropy	entropy	entropy
max_depth	6	4	8	10
min_samples_leaf	3	3	7	2
min_samples_split	4	3	2	5
random_state	0	0	0	0

Tabla A.35: Tabla de parámetros para Árbol de decisión sobre Fashion MNIST.

Los resultados obtenidos por el Árbol de decisión se presentan en la Tabla A.36.

Ejemplos/Clase: 10	
Exactitud	0.463
Precisión	0.484
Recall	0.463
F1	0.462
MCC	0.406

Ejemplos/Clase: 50	
Exactitud	0.649
Precisión	0.618
Recall	0.649
F1	0.618
MCC	0.617

Ejemplos/Clase: 250	
Exactitud	0.729
Precisión	0.731
Recall	0.729
F1	0.729
MCC	0.699

Ejemplos/Clase: 500	
Exactitud	0.756
Precisión	0.755
Recall	0.756
F1	0.754
MCC	0.729

Tabla A.36: Tabla de resultados de Árbol de decisión sobre Fashion MNIST.

A.2.2. Bosque Aleatorio

Los parámetros utilizados para la implementación del Bosque aleatorio se presentan en la Tabla A.37.

	Ejemplos por clase			
Parámetros	10	50	250	500
criterion	gini	gini	entropy	entropy
min_samples_leaf	1	1	2	2
min_samples_split	2	2	2	6
n_estimators	5000	250	250	250
random_state	0	0	0	0

Tabla A.37: Tabla de parámetros para Bosque Aleatorio sobre Fashion MNIST.

Los resultados obtenidos por el algoritmo Bosque aleatorio se presentan en la Tabla A.38.

Ejemplos/Clase: 10	
Exactitud	0.675
Precisión	0.674
Recall	0.675
F1	0.673
MCC	0.639

Ejemplos/Clase: 50	
Exactitud	0.778
Precisión	0.774
Recall	0.778
F1	0.775
MCC	0.754

Ejemplos/Clase: 250	
Exactitud	0.822
Precisión	0.82
Recall	0.822
F1	0.820
MCC	0.803

Ejemplos/Clase: 500	
Exactitud	0.837
Precisión	0.835
Recall	0.837
F1	0.834
MCC	0.819

Tabla A.38: Tabla de resultados de Bosque Aleatorio sobre Fashion MNIST.

A.2.3. Máquina de Soporte de Vectores

Los parámetros utilizados para la implementación de la Máquina de Soporte de Vectores se presentan en la Tabla A.39.

	Ejemplos por clase			
Parámetros	10	50	250	500
C	3	5	5	5
degree	1	1	1	1
kernel	rbf	rbf	rbf	rbf
random_state	0	0	0	0

Tabla A.39: Tabla de parámetros para la Máquina de Soporte de Vectores sobre Fashion MNIST.

Los resultados obtenidos por la Máquina de Soporte de Vectores se presentan en la Tabla A.40.

Ejemplos/Clase: 10	
Exactitud	0.685
Precisión	0.685
Recall	0.685
F1	0.682
MCC	0.65

Ejemplos/Clase: 50	
Exactitud	0.798
Precisión	0.799
Recall	0.798
F1	0.798
MCC	0.775

Ejemplos/Clase: 250	
Exactitud	0.837
Precisión	0.837
Recall	0.837
F1	0.837
MCC	0.819

Ejemplos/Clase: 500	
Exactitud	0.851
Precisión	0.85
Recall	0.851
F1	0.850
MCC	0.835

Tabla A.40: Tabla de resultados de la Máquina de Soporte de Vectores sobre Fashion MNIST.

A.2.4. Red Neuronal Profunda

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.41.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	50	18	17	17
Tamaño Lote	32			
Tasa de Aprendizaje	0.0005			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.41: Tabla de parámetros de la Red Neuronal Profunda sobre Fashion MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.42.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	28x28x1				
Convolucional		(7,7)	64		ReLU
AgrupaciónMáxima		(2,2)			
Convolucional		(5,5)	128		ReLU
AgrupaciónMáxima		(2,2)			ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	128		ReLU
Convolucional		(5,5)	128		ReLU
Aplanar					
Densa				64	ReLU
Densa				64	ReLU
Densa				64	ReLU
Densa				10	Softmax

Tabla A.42: Tabla resumen de la Estructura de Red Neuronal Profunda sobre Fashion MNIST.

Los resultados obtenidos por la Red Neuronal Profunda se presentan en la Tabla A.43.

Ejemplos/Clase: 10	
Exactitud	0.624
Precisión	0.633
Recall	0.624
F1	0.622
MCC	0.584

Ejemplos/Clase: 50	
Exactitud	0.762
Precisión	0.78
Recall	0.762
F1	0.754
MCC	0.739

Ejemplos/Clase: 250	
Exactitud	0.802
Precisión	0.838
Recall	0.802
F1	0.803
MCC	0.784

Ejemplos/Clase: 500	
Exactitud	0.854
Precisión	0.854
Recall	0.854
F1	0.854
MCC	0.838

Tabla A.43: Resultados de la Red Neuronal Profunda sobre Fashion MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.10.

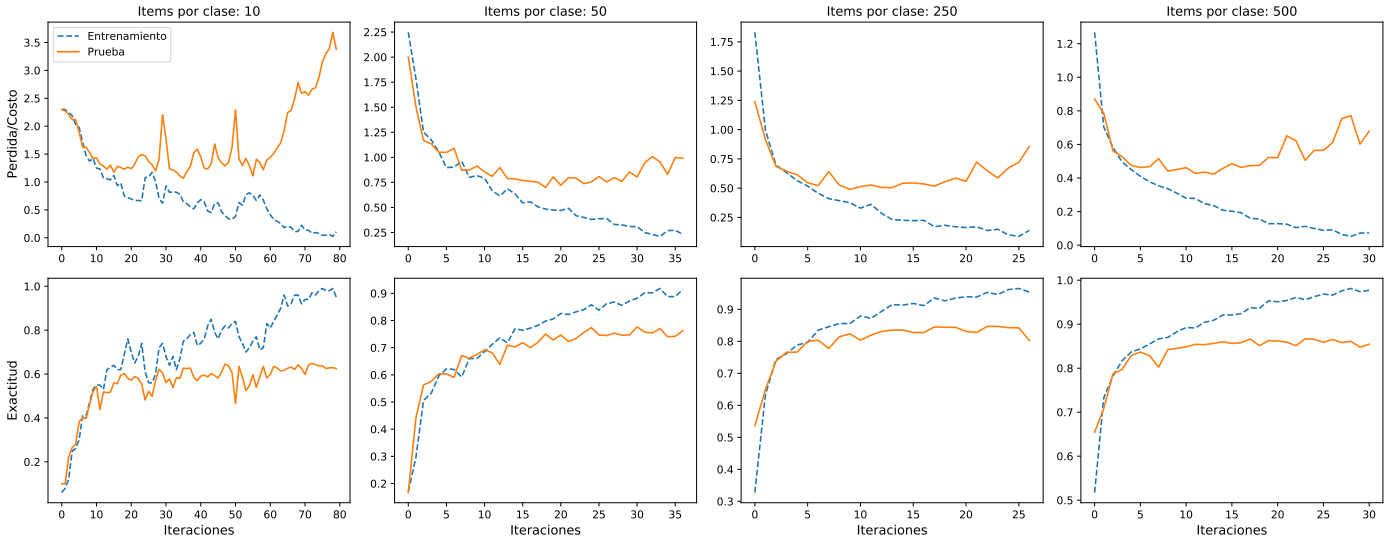

Figura A.10: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda sobre Fashion MNIST.

A.2.5. Red Neuronal Profunda con Dropout

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.44.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	20	25	25	25
Tamaño Lote	24	32		
Tasa de Aprendizaje	0.0001			
Limite de iteraciones (Entrenamiento)	200			

Tabla A.44: Tabla de parámetros para la Red Neuronal Profunda con Dropout sobre Fashion MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.45.

Capa	Tamaño	Kernel	Filtros	Probabilidad	Unidades	Activación
Entrada	28x28x1					
Convolucional		(7,7)	64			ReLU
AgrupaciónMáxima		(2,2)				
Convolucional		(5,5)	128			ReLU
AgrupaciónMáxima		(2,2)				ReLU
Convolucional		(5,5)	64			ReLU
Convolucional		(5,5)	64			ReLU
Convolucional		(5,5)	128			ReLU
Convolucional		(5,5)	128			ReLU
Aplanar						
Dropout				0.5		ReLU
Densa					64	
Dropout				0.5		ReLU
Densa					64	
Dropout				0.25		ReLU
Densa					64	
Dropout				0.25		
Densa					10	Softmax

Tabla A.45: Tabla resumen de la Estructura de Red Neuronal Profunda con Dropout sobre Fashion MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con Dropout se presentan en la Tabla A.46.

Ejemplos/Clase: 10	
Exactitud	0.617
Precisión	0.619
Recall	0.617
F1	0.592
MCC	0.58

Ejemplos/Clase: 50	
Exactitud	0.718
Precisión	0.732
Recall	0.718
F1	0.718
MCC	0.688

Ejemplos/Clase: 250	
Exactitud	0.844
Precisión	0.841
Recall	0.844
F1	0.841
MCC	0.827

Ejemplos/Clase: 500	
Exactitud	0.870
Precisión	0.872
Recall	0.87
F1	0.870
MCC	0.856

Tabla A.46: Resultados de la Red Neuronal Profunda con Dropout sobre Fashion MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.11.

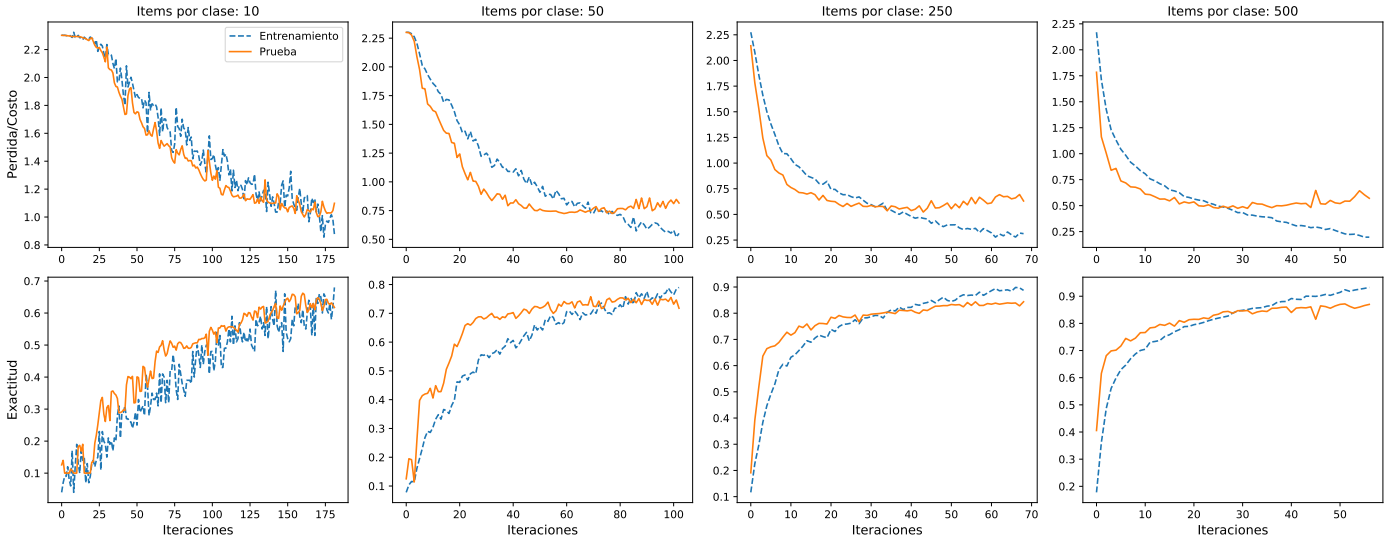

Figura A.11: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Dropout sobre Fashion MNIST.

A.2.6. Red Neuronal Profunda con capa Agrupación Promedio Global

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.47.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	30	18	17	17
Tamaño Lote	20	32		
Tasa de Aprendizaje	0.0005			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.47: Tabla de parámetros para la Red Neuronal Profunda con capa Agrupación Promedio Global sobre Fashion MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.48.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	28x28x1				
Convolution		(7,7)	64		ReLU
AgrupaciónMáxima		(2,2)			
Convolution		(5,5)	128		ReLU
AgrupaciónMáxima		(2,2)			
Convolution		(5,5)	64		ReLU
Convolution		(5,5)	64		ReLU
Convolution		(5,5)	128		ReLU
Convolution		(5,5)	128		
AgrupaciónPromedioGlobal					
Densa				10	Softmax

Tabla A.48: Tabla resumen de la Estructura de Red Neuronal Profunda con capa Agrupación Promedio Global sobre Fashion MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con capa de Agrupación Promedio Global se presentan en la Tabla A.49.

Ejemplos/Clase: 10	
Exactitud	0.683
Precisión	0.681
Recall	0.683
F1	0.679
MCC	0.649

Ejemplos/Clase: 50	
Exactitud	0.792
Precisión	0.796
Recall	0.792
F1	0.790
MCC	0.771

Ejemplos/Clase: 250	
Exactitud	0.854
Precisión	0.857
Recall	0.854
F1	0.853
MCC	0.838

Ejemplos/Clase: 500	
Exactitud	0.846
Precisión	0.858
Recall	0.846
F1	0.845
MCC	0.83

Tabla A.49: Resultados de la Red Neuronal Profunda con capa Agrupación Promedio Global sobre Fashion MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.12.

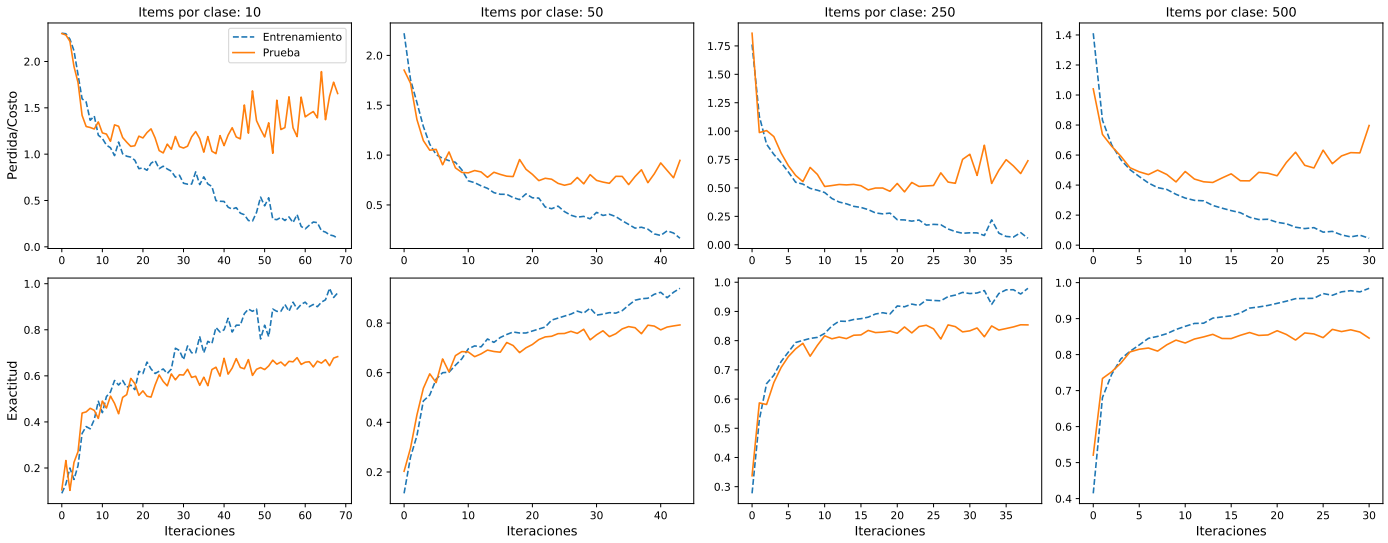

Figura A.12: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con capa Agrupación Promedio Global sobre Fashion MNIST.

A.2.7. Red Neuronal Profunda con Normalización por lotes

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.50.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	20	16	17	17
Tamaño Lote	20	32		
Tasa de Aprendizaje	0.001			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.50: Tabla de parámetros para Red Neuronal Profunda con Normalización por Lotes sobre Fashion MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.51.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	28x28x1				
Convolucional		(7,7)	64		ReLU
NormalizaciónPorLotes					
AgrupaciónMáxima		(2,2)			
Convolucional		(5,5)	128		ReLU
AgrupaciónMáxima		(2,2)			
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	128		ReLU
Convolucional		(5,5)	128		ReLU
Aplanar					
Densa				64	ReLU
Densa				64	ReLU
Densa				64	ReLU
Densa				10	Softmax

Tabla A.51: Tabla resumen de la Estructura de Red Neuronal Profunda con Normalización por Lotes sobre Fashion MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con capa de Normalización por Lotes se presentan en la Tabla A.52.

Ejemplos/Clase: 10	
Exactitud	0.613
Precisión	0.63
Recall	0.613
F1	0.611
MCC	0.573

Ejemplos/Clase: 50	
Exactitud	0.781
Precisión	0.794
Recall	0.781
F1	0.770
MCC	0.762

Ejemplos/Clase: 250	
Exactitud	0.816
Precisión	0.828
Recall	0.816
F1	0.815
MCC	0.797

Ejemplos/Clase: 500	
Exactitud	0.866
Precisión	0.868
Recall	0.866
F1	0.866
MCC	0.851

Tabla A.52: Resultados de la Red Neuronal Profunda con Normalización por Lotes sobre Fashion MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.13.

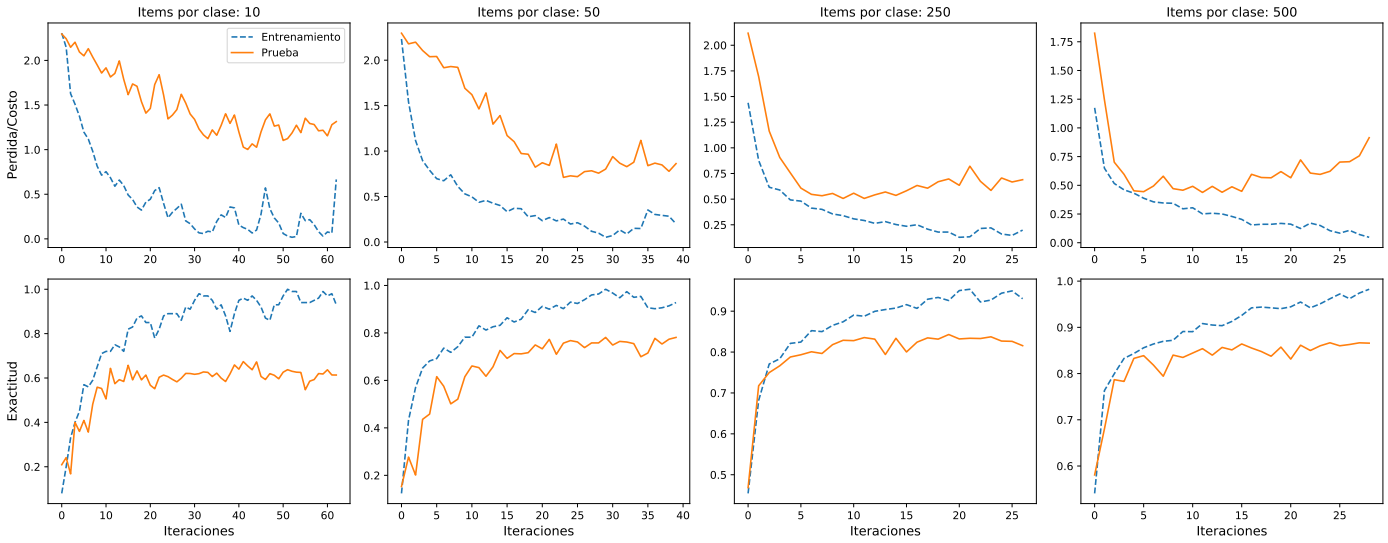

Figura A.13: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Normalización por Lotes sobre Fashion MNIST.

A.2.8. Red Neuronal Profunda con función de costo Similitud de Coseno

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.53.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	15	8	6	4
Tamaño Lote	20	32		
Tasa de Aprendizaje	0.0005			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.53: Tabla de parámetros para Red Neuronal Profunda con función de costo Similitud de Coseno sobre Fashion MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.54.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	28x28x1				
Convolucional		(7,7)	64		ReLU
AgrupaciónMáxima		(2,2)			
Convolucional		(5,5)	128		ReLU
AgrupaciónMáxima		(2,2)			ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	128		ReLU
Convolucional		(5,5)	128		ReLU
Aplanar					
Densa				64	ReLU
Densa				64	ReLU
Densa				64	ReLU
Densa				10	Softmax

Tabla A.54: Tabla resumen de la Estructura de Red Neuronal Profunda con función de costo Similitud de Coseno sobre Fashion MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con función de costo Similitud de Coseno se presentan en la Tabla A.55.

Ejemplos/Clase: 10	
Exactitud	0.631
Precisión	0.643
Recall	0.631
F1	0.629
MCC	0.591

Ejemplos/Clase: 50	
Exactitud	0.743
Precisión	0.746
Recall	0.743
F1	0.735
MCC	0.717

Ejemplos/Clase: 250	
Exactitud	0.803
Precisión	0.811
Recall	0.803
F1	0.803
MCC	0.782

Ejemplos/Clase: 500	
Exactitud	0.855
Precisión	0.862
Recall	0.855
F1	0.856
MCC	0.839

Tabla A.55: Resultados de la Red Neuronal Profunda con función de costo Similitud de Coseno sobre Fashion MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.14.

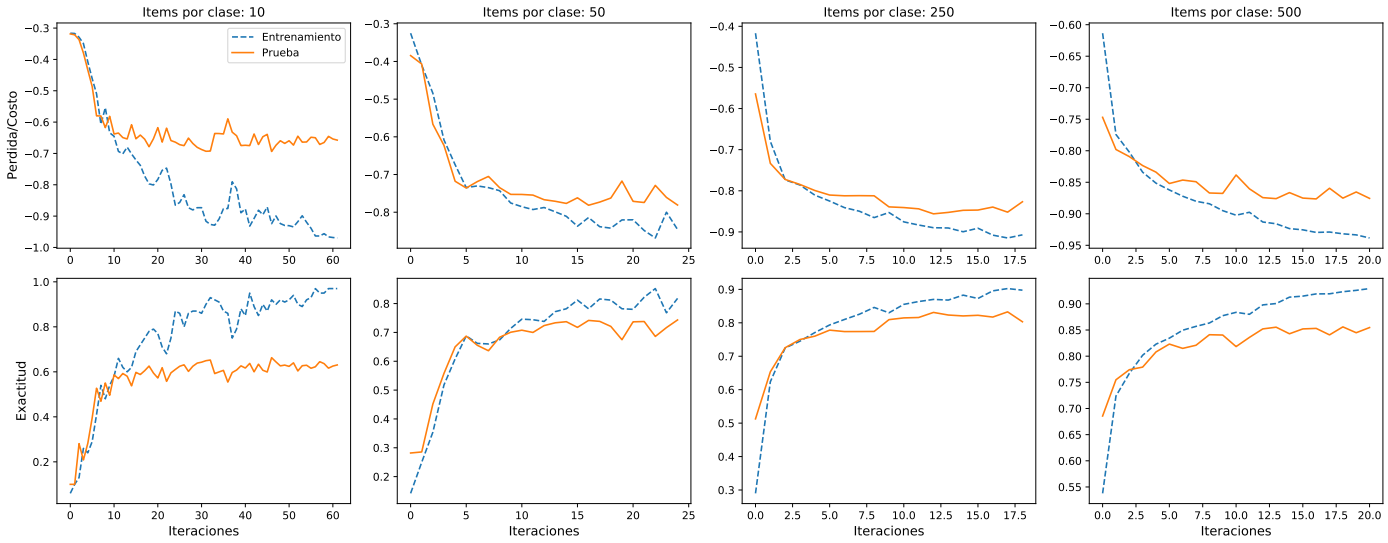

Figura A.14: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con función de costo Similitud de Coseno sobre Fashion MNIST.

A.2.9. Red Neuronal Profunda con Convolución Dilatada

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.56.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	20	18	16	15
Tamaño Lote	24	32		
Tasa de Aprendizaje	0.0005			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.56: Tabla de parámetros para Red Neuronal Profunda con Convolución Dilatada sobre Fashion MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.57.

Capa	Tamaño	Kernel	Filtros	Tasa de Dilatación	Unidades	Activación
Entrada	28x28x1					
Convolucional		(7,7)	64	2		ReLU
AgrupaciónMáxima		(2,2)				
Convolucional		(5,5)	128	2		ReLU
AgrupaciónMáxima		(2,2)				ReLU
Convolucional		(5,5)	64	2		ReLU
Convolucional		(5,5)	64	2		ReLU
Convolucional		(5,5)	128	2		ReLU
Convolucional		(5,5)	128	2		ReLU
Aplanar						
Densa					64	ReLU
Densa					64	ReLU
Densa					64	ReLU
Densa					10	Softmax

Tabla A.57: Tabla resumen de la Estructura de Red Neuronal Profunda con Convolución Dilatada sobre Fashion MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con Convolución Dilatada se presentan en la Tabla A.58.

Ejemplos/Clase: 10	
Exactitud	0.653
Precisión	0.639
Recall	0.653
F1	0.638
MCC	0.617

Ejemplos/Clase: 50	
Exactitud	0.780
Precisión	0.78
Recall	0.78
F1	0.777
MCC	0.757

Ejemplos/Clase: 250	
Exactitud	0.837
Precisión	0.84
Recall	0.837
F1	0.836
MCC	0.819

Ejemplos/Clase: 500	
Exactitud	0.856
Precisión	0.854
Recall	0.856
F1	0.854
MCC	0.84

Tabla A.58: Resultados de la Red Neuronal Profunda con Convolución Dilatada sobre Fashion MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.15.

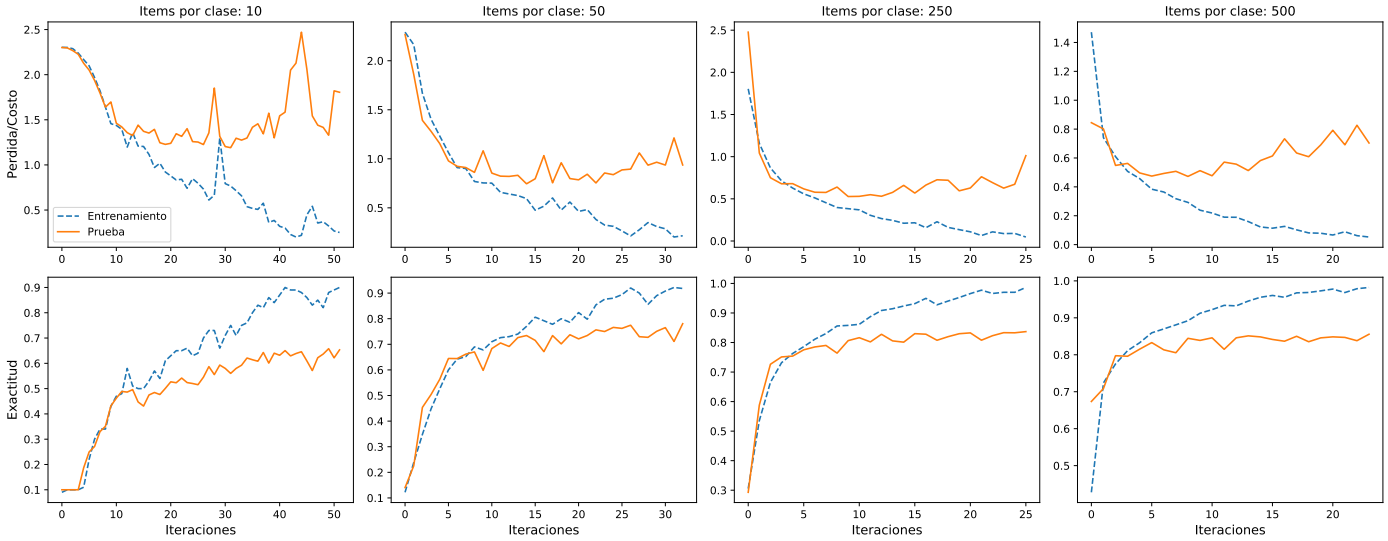

Figura A.15: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Convolución Dilatada sobre Fashion MNIST.

A.2.10. Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.59.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	50	18	17	17
Tamaño Lote	32			
Tasa de Aprendizaje mínima	0.0001			
Tasa de Aprendizaje máxima	0.001			
Delta (Decadencia cíclica de tasa de aprendizaje)	10			
s (Decadencia cíclica de tasa de aprendizaje)	10			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.59: Tabla de parámetros para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre Fashion MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.60.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	28x28x1				
Convolucional		(7,7)	64		ReLU
AgrupaciónMáxima		(2,2)			
Convolucional		(5,5)	128		ReLU
AgrupaciónMáxima		(2,2)			ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	64		ReLU
Convolucional		(5,5)	128		ReLU
Convolucional		(5,5)	128		ReLU
Aplanar					
Densa				64	ReLU
Densa				64	ReLU
Densa				64	ReLU
Densa				10	Softmax

Tabla A.60: Tabla resumen de la Estructura de Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre Fashion MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con Decadencia Cíclica de Tasa de

Aprendizaje se presentan en la Tabla A.61.

Ejemplos/Clase: 10	
Exactitud	0.630
Precisión	0.633
Recall	0.63
F1	0.629
MCC	0.589

Ejemplos/Clase: 50	
Exactitud	0.738
Precisión	0.74
Recall	0.738
F1	0.734
MCC	0.71

Ejemplos/Clase: 250	
Exactitud	0.827
Precisión	0.835
Recall	0.827
F1	0.829
MCC	0.808

Ejemplos/Clase: 500	
Exactitud	0.851
Precisión	0.851
Recall	0.851
F1	0.850
MCC	0.834

Tabla A.61: Resultados de la Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre Fashion MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.16.

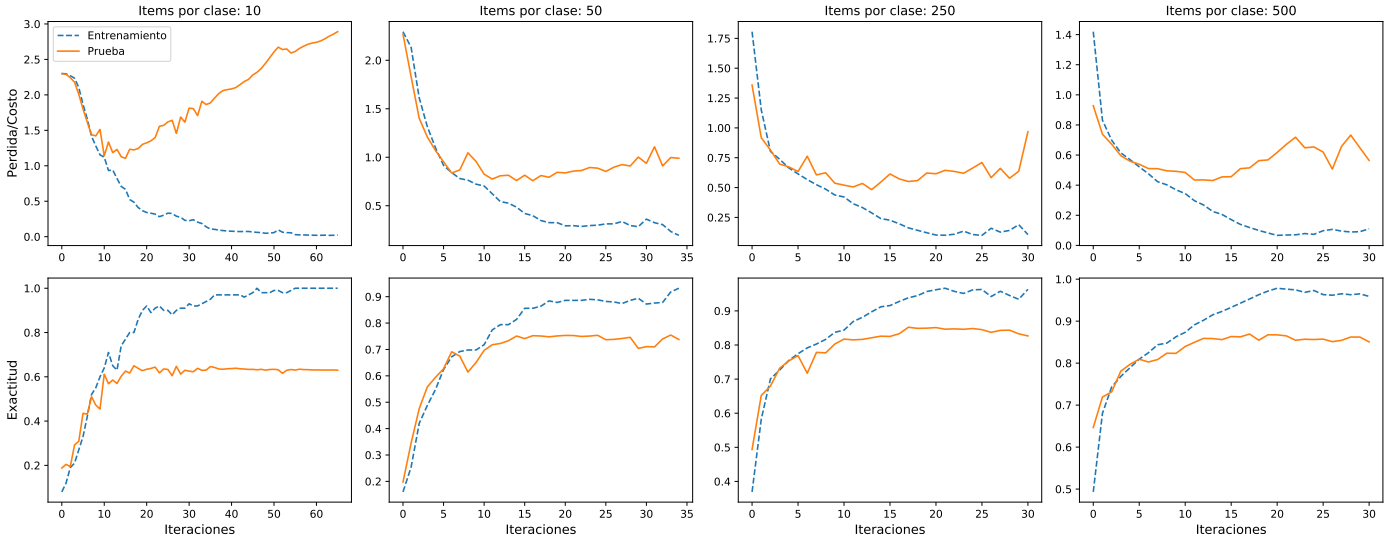

Figura A.16: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre Fashion MNIST.

A.2.11. Red Neuronal Profunda con combinación de técnicas (C1)

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.62.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	75	30	30	30
Tamaño Lote	20	32		
Tasa de Aprendizaje mínima	0.00005			
Tasa de Aprendizaje máxima	0.0035			
Delta (Decadencia cíclica de tasa de aprendizaje)	10			
s (Decadencia cíclica de tasa de aprendizaje)	10			
Limite de iteraciones (Entrenamiento)	350			

Tabla A.62: Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas sobre Fashion MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.63.

Capa	Tamaño	Kernel	Filtros	Tasa de Dilatación	Probabilidad	Unidades	Activación
Entrada	28x28x1						
Convolucional		(7,7)	64	2			ReLU
NormalizaciónPorLotes							
AgrupaciónMáxima		(2,2)					
Convolucional		(5,5)	128	2			ReLU
NormalizaciónPorLotes							
AgrupaciónMáxima		(2,2)					
Convolucional		(5,5)	64	2			ReLU
NormalizaciónPorLotes							
Convolucional		(5,5)	64	2			ReLU
Convolucional		(5,5)	128	2			ReLU
Convolucional		(5,5)	128	1			ReLU
AgrupaciónPromedioGlobal							
Dropout					0.1		
Densa						10	Softmax

Tabla A.63: Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas sobre Fashion MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con combinación de técnicas se presentan en la Tabla A.64.

Ejemplos/Clase: 10	
Exactitud	0.676
Precisión	0.683
Recall	0.676
F1	0.678
MCC	0.641

Ejemplos/Clase: 50	
Exactitud	0.781
Precisión	0.789
Recall	0.781
F1	0.783
MCC	0.757

Ejemplos/Clase: 250	
Exactitud	0.813
Precisión	0.822
Recall	0.813
F1	0.812
MCC	0.793

Ejemplos/Clase: 500	
Exactitud	0.843
Precisión	0.851
Recall	0.843
F1	0.845
MCC	0.827

Tabla A.64: Resultados de la Red Neuronal Profunda con combinación de técnicas sobre Fashion MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.17.

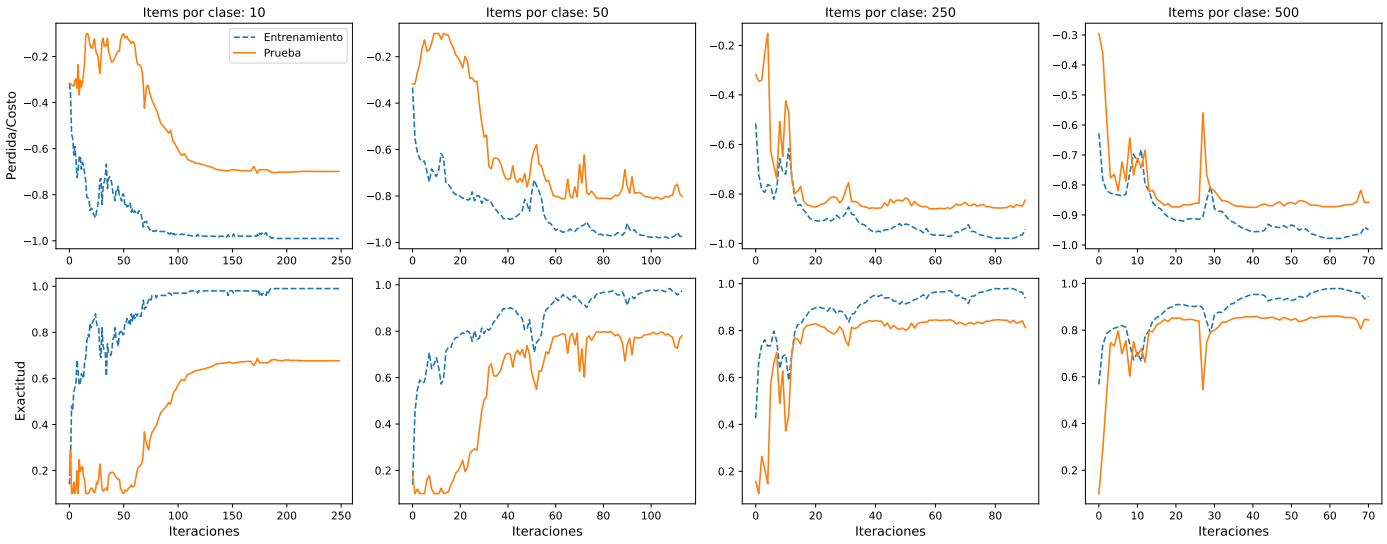

Figura A.17: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas sobre Fashion MNIST.

A.2.12. Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos (C2)

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.65.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	20	20	20	15
Tamaño Lote	24	32		
Interpolación	Bilineal			
Rango de rotación (Aumento de datos)	0º a 25º			
Desplazamiento horizontal (Aumento de datos)	10 %			
Desplazamiento vertical (Aumento de datos)	10 %			
Rango de acercamiento (Aumento de datos)	85 % a 100 %			
Volteo horizontal (Aumento de datos)	Si			
Tasa de Aprendizaje mínima	0.0001			
Tasa de Aprendizaje máxima	0.001			
Delta (Decadencia cíclica de tasa de aprendizaje)	10			
s (Decadencia cíclica de tasa de aprendizaje)	10			
Limite de iteraciones (Entrenamiento)	100			

Tabla A.65: Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre Fashion MNIST.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.66.

Capa	Tamaño	Tasa de Aumento	Probabilidad	Unidades	Activación
Entrada	28x28x3				
Ampliar		(8,8)			
EfficientNetB0					
AgrupaciónPromedioGlobal					
NormalizaciónPorLotes					
Dropout			0.5		
Densa				10	Softmax

Tabla A.66: Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre Fashion MNIST.

Los resultados obtenidos por la Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos se presentan en la Tabla A.67.

Ejemplos/Clase: 10	
Exactitud	0.655
Precisión	0.683
Recall	0.655
F1	0.644
MCC	0.623

Ejemplos/Clase: 50	
Exactitud	0.784
Precisión	0.803
Recall	0.784
F1	0.786
MCC	0.761

Ejemplos/Clase: 250	
Exactitud	0.834
Precisión	0.843
Recall	0.834
F1	0.833
MCC	0.817

Ejemplos/Clase: 500	
Exactitud	0.843
Precisión	0.854
Recall	0.843
F1	0.845
MCC	0.827

Tabla A.67: Resultados de la Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre Fashion MNIST.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.18.

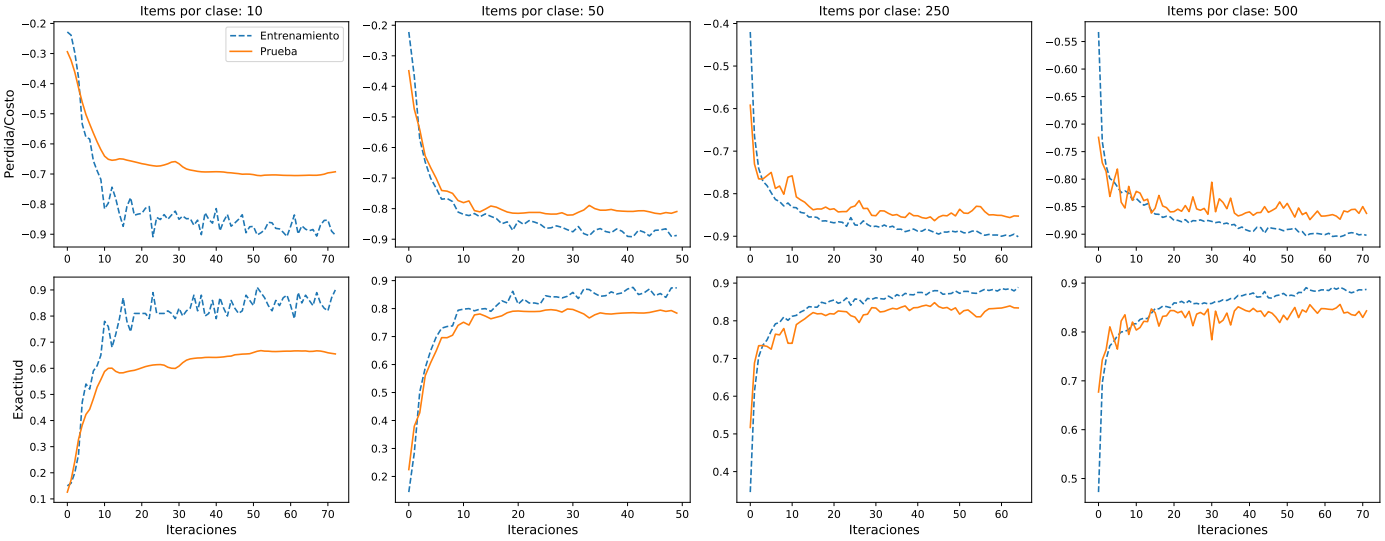

Figura A.18: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos sobre Fashion MNIST.

A.2.13. Ensamblaje de todas las Redes Neuronales Profundas

Los resultados obtenidos por el ensamblaje de redes neuronales se presentan en la Tabla A.68.

Ejemplos/Clase: 10	
Exactitud	0.702
Precisión	0.699
Recall	0.702
F1	0.695
MCC	0.67

Ejemplos/Clase: 50	
Exactitud	0.819
Precisión	0.814
Recall	0.819
F1	0.815
MCC	0.799

Ejemplos/Clase: 250	
Exactitud	0.874
Precisión	0.874
Recall	0.874
F1	0.874
MCC	0.86

Ejemplos/Clase: 500	
Exactitud	0.890
Precisión	0.890
Recall	0.890
F1	0.890
MCC	0.878

Tabla A.68: Resultados del ensamblaje de todas las Redes Neuronales Profundas sobre Fashion MNIST.

A.3. Configuración de algoritmos entrenados sobre subconjuntos de CIFAR-10

A.3.1. Árbol de decisión

Los parámetros utilizados para la implementación del Árbol de decisión se presentan en la Tabla A.69.

	Ejemplos por clase			
Parámetros	10	50	250	500
criterion	entropy	entropy	gini	gini
max_depth	4	10	6	8
min_samples_leaf	5	1	7	2
min_samples_split	6	5	5	6
random_state	0	0	0	0

Tabla A.69: Tabla de parámetros para Árbol de decisión sobre CIFAR-10.

Los resultados obtenidos por el Árbol de decisión se presentan en la Tabla A.70.

Ejemplos/Clase: 10	
Exactitud	0.164
Precisión	0.184
Recall	0.164
F1	0.165
MCC	0.071

Ejemplos/Clase: 50	
Exactitud	0.192
Precisión	0.196
Recall	0.192
F1	0.193
MCC	0.102

Ejemplos/Clase: 250	
Exactitud	0.229
Precisión	0.238
Recall	0.229
F1	0.217
MCC	0.146

Ejemplos/Clase: 500	
Exactitud	0.252
Precisión	0.254
Recall	0.252
F1	0.246
MCC	0.17

Tabla A.70: Tabla de resultados de Árbol de decisión sobre CIFAR-10.

A.3.2. Bosque Aleatorio

Los parámetros utilizados para la implementación del Bosque Aleatorio se presentan en la Tabla A.71.

	Ejemplos por clase			
Parámetros	10	50	250	500
criterion	entropy	gini	gini	gini
min_samples_leaf	1	1	1	1
min_samples_split	4	1	3	5
n_estimators	2500	250	500	2500
random_state	0	0	0	0

Tabla A.71: Tabla de parámetros para Bosque Aleatorio sobre CIFAR-10.

Los resultados obtenidos por el Bosque Aleatorio se presentan en la Tabla A.72.

Ejemplos/Clase: 10	
Exactitud	0.284
Precisión	0.294
Recall	0.284
F1	0.272
MCC	0.207

Ejemplos/Clase: 50	
Exactitud	0.334
Precisión	0.331
Recall	0.334
F1	0.329
MCC	0.261

Ejemplos/Clase: 250	
Exactitud	0.403
Precisión	0.396
Recall	0.403
F1	0.396
MCC	0.337

Ejemplos/Clase: 500	
Exactitud	0.425
Precisión	0.418
Recall	0.425
F1	0.417
MCC	0.362

Tabla A.72: Tabla de resultados de Bosque Aleatorio sobre CIFAR-10.

A.3.3. Máquina de Soporte de Vectores

Los parámetros utilizados para la implementación de la Máquina de Soporte de Vectores se presentan en la Tabla A.73.

	Ejemplos por clase			
Parámetros	10	50	250	500
C	5	5	3	3
degree	1	1	1	1
kernel	rbf	rbf	rbf	rbf
random_state	0	0	0	0

Tabla A.73: Tabla de parámetros para Máquina de Soporte de Vectores sobre CIFAR-10.

Los resultados obtenidos por la Máquina de Soporte de Vectores se presentan en la Tabla

A.74.

Ejemplos/Clase: 10	
Exactitud	0.262
Precisión	0.279
Recall	0.262
F1	0.262
MCC	0.181

Ejemplos/Clase: 50	
Exactitud	0.339
Precisión	0.348
Recall	0.339
F1	0.340
MCC	0.266

Ejemplos/Clase: 250	
Exactitud	0.422
Precisión	0.42
Recall	0.422
F1	0.419
MCC	0.357

Ejemplos/Clase: 500	
Exactitud	0.456
Precisión	0.456
Recall	0.456
F1	0.455
MCC	0.396

Tabla A.74: Tabla de resultados de Máquina de Soporte de Vectores sobre CIFAR-10.

A.3.4. Red Neuronal Profunda

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.75.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	55	30	10	10
Tamaño Lote	20	32		
Tasa de Aprendizaje	0.00005			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.75: Tabla de parámetros para Red Neuronal Profunda sobre CIFAR-10.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.76.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	32x32x3				
Convolucional		(7,7)	128		ReLU
AgrupaciónMáxima		(4,4)			
Convolucional		(7,7)	128		ReLU
Convolucional		(7,7)	64		ReLU
Convolucional		(7,7)	64		ReLU
Aplanar					
Densa				48	ReLU
Densa				32	ReLU
Densa				10	Softmax

Tabla A.76: Tabla resumen de la Estructura de la Red Neuronal Profunda sobre CIFAR-10.

Los resultados obtenidos por la Red Neuronal Profunda se presentan en la Tabla A.77.

Ejemplos/Clase: 10	
Exactitud	0.285
Precisión	0.285
Recall	0.285
F1	0.278
MCC	0.206

Ejemplos/Clase: 50	
Exactitud	0.357
Precisión	0.377
Recall	0.357
F1	0.357
MCC	0.288

Ejemplos/Clase: 250	
Exactitud	0.489
Precisión	0.504
Recall	0.489
F1	0.483
MCC	0.435

Ejemplos/Clase: 500	
Exactitud	0.575
Precisión	0.571
Recall	0.575
F1	0.569
MCC	0.528

Tabla A.77: Resultados de la Red Neuronal Profunda sobre CIFAR-10.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.19.

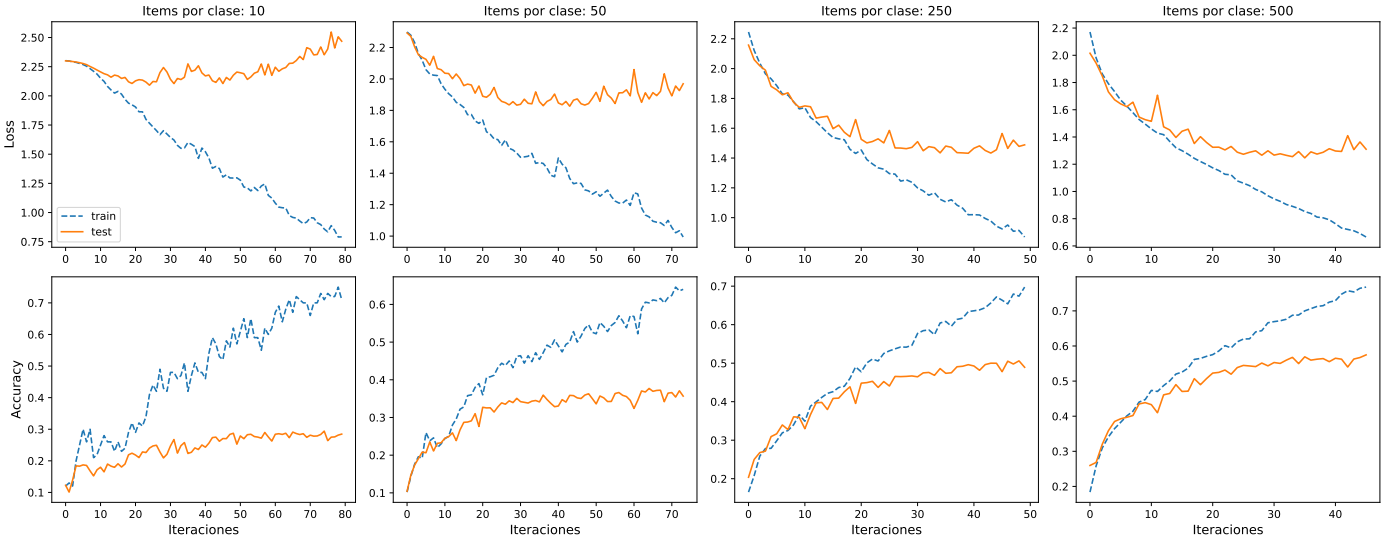

Figura A.19: Gráficas de costo (Loss) y Exactitud (Accuracy) durante entrenamiento para Red Neuronal Profunda sobre CIFAR-10.

A.3.5. Red Neuronal Profunda con Dropout

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.78.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	180	100	60	40
Tamaño Lote	20		32	
Tasa de Aprendizaje	0.00005			
Limite de iteraciones (Entrenamiento)	250			

Tabla A.78: Tabla de parámetros para Red Neuronal Profunda con Dropout sobre CIFAR-10.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.79.

Capa	Tamaño	Kernel	Filtros	Probabilidad	Unidades	Activación
Entrada	32x32x3					
Convolucional		(7,7)	128			ReLU
AgrupaciónMáxima		(4,4)				
Convolucional		(7,7)	128			ReLU
Convolucional		(7,7)	64			ReLU
Convolucional		(7,7)	64			ReLU
Aplanar						
Dropout				0.25		
Densa					48	ReLU
Dropout				0.25		
Densa					32	ReLU
Dropout				0.1		
Densa					10	Softmax

Tabla A.79: Tabla resumen de la Estructura de la Red Neuronal Profunda con Dropout sobre CIFAR-10.

Los resultados obtenidos por la Red Neuronal Profunda con Dropout se presentan en la Tabla A.80.

Ejemplos/Clase: 10	
Exactitud	0.294
Precisión	0.3
Recall	0.294
F1	0.292
MCC	0.216

Ejemplos/Clase: 50	
Exactitud	0.389
Precisión	0.411
Recall	0.389
F1	0.393
MCC	0.322

Ejemplos/Clase: 250	
Exactitud	0.524
Precisión	0.526
Recall	0.524
F1	0.519
MCC	0.473

Ejemplos/Clase: 500	
Exactitud	0.604
Precisión	0.604
Recall	0.604
F1	0.601
MCC	0.561

Tabla A.80: Resultados para Red Neuronal Profunda con Dropout sobre CIFAR-10.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.20.

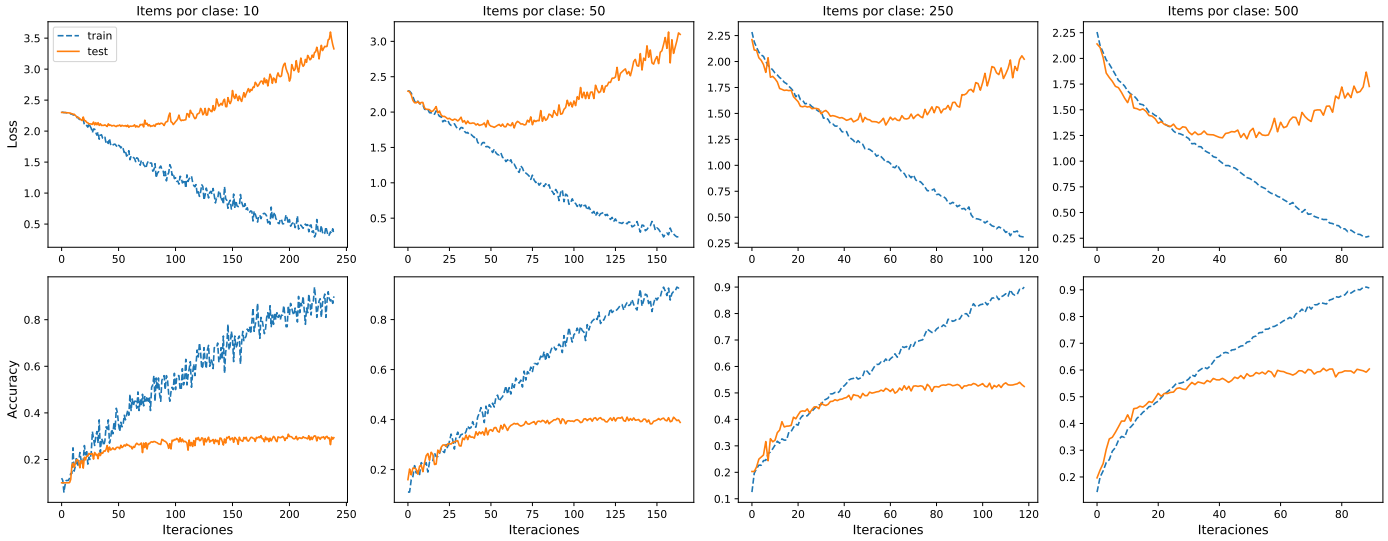

Figura A.20: Gráficas de costo (Loss) y Exactitud (Accuracy) durante entrenamiento para Red Neuronal Profunda con Dropout sobre CIFAR-10.

A.3.6. Red Neuronal Profunda con capa Agrupación Promedio Global

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.81.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	55	20	20	20
Tamaño Lote	20	32	64	
Tasa de Aprendizaje	0.00005			
Limite de iteraciones (Entrenamiento)	120			

Tabla A.81: Tabla de parámetros para la Red Neuronal Profunda con capa Agrupación Promedio Global sobre CIFAR-10.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.82.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	28x28x1				
Convolucional		(7,7)	128		ReLU
AgrupaciónMáxima		(4,4)			
Convolucional		(7,7)	128		ReLU
Convolucional		(7,7)	64		ReLU
Convolucional		(7,7)	64		ReLU
AgrupaciónPromedioGlobal					
Densa				10	Softmax

Tabla A.82: Tabla resumen de la Estructura de Red Neuronal Profunda con capa Agrupación Promedio Global sobre CIFAR-10.

Los resultados obtenidos por la Red Neuronal Profunda con capa de Agrupación Promedio Global se presentan en la Tabla A.83.

Ejemplos/Clase: 10	
Exactitud	0.293
Precisión	0.283
Recall	0.293
F1	0.277
MCC	0.217

Ejemplos/Clase: 50	
Exactitud	0.390
Precisión	0.391
Recall	0.39
F1	0.371
MCC	0.326

Ejemplos/Clase: 250	
Exactitud	0.498
Precisión	0.487
Recall	0.498
F1	0.475
MCC	0.445

Ejemplos/Clase: 500	
Exactitud	0.590
Precisión	0.609
Recall	0.59
F1	0.590
MCC	0.546

Tabla A.83: Resultados de la Red Neuronal Profunda con capa Agrupación Promedio Global sobre CIFAR-10.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.21.

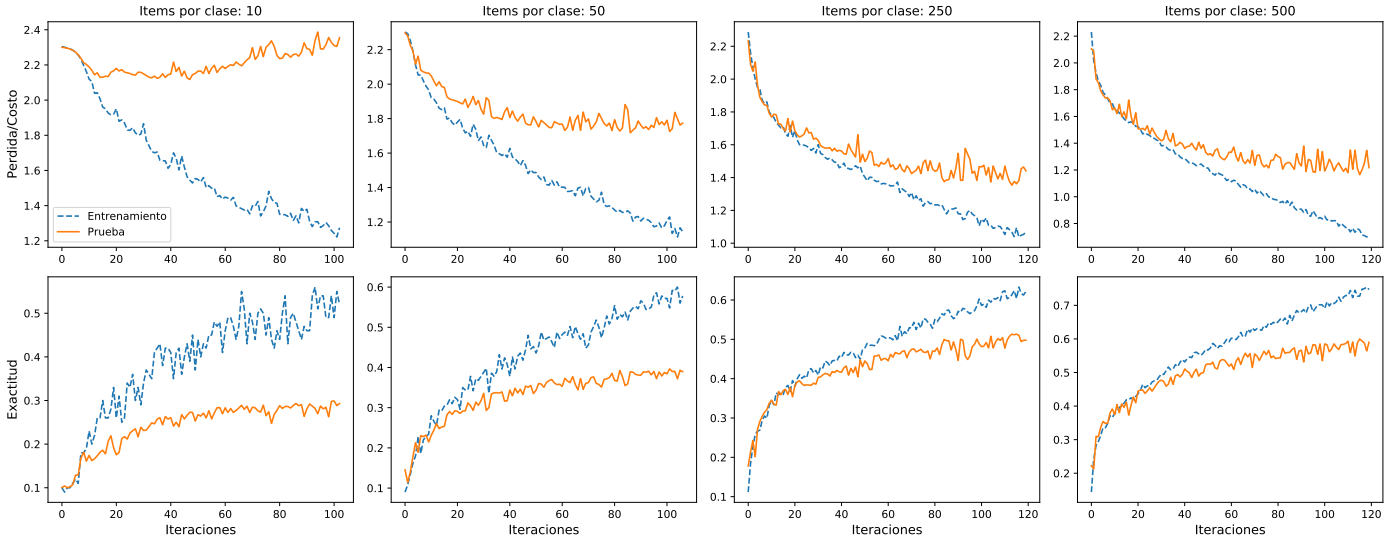

Figura A.21: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con capa Agrupación Promedio Global sobre CIFAR-10.

A.3.7. Red Neuronal Profunda con Normalización por lotes

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.84.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	15	15	15	15
Tamaño Lote	20	28	32	
Tasa de Aprendizaje	0.00005			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.84: Tabla de parámetros para Red Neuronal Profunda con Normalización por Lotes sobre CIFAR-10.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.85.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	32x32x3				
Convolucional		(7,7)	128		ReLU
NormalizaciónPorLotes					
AgrupaciónMáxima		(4,4)			
Convolucional		(7,7)	128		ReLU
Convolucional		(7,7)	64		ReLU
Convolucional		(7,7)	64		ReLU
Aplanar					
Densa				48	ReLU
Densa				32	ReLU
Densa				10	Softmax

Tabla A.85: Tabla resumen de la Estructura de Red Neuronal Profunda con Normalización por Lotes sobre CIFAR-10.

Los resultados obtenidos por la Red Neuronal Profunda con capa de Normalización por Lotes se presentan en la Tabla A.86.

Ejemplos/Clase: 10	
Exactitud	0.295
Precisión	0.302
Recall	0.295
F1	0.291
MCC	0.217

Ejemplos/Clase: 50	
Exactitud	0.410
Precisión	0.417
Recall	0.41
F1	0.410
MCC	0.345

Ejemplos/Clase: 250	
Exactitud	0.512
Precisión	0.519
Recall	0.512
F1	0.507
MCC	0.46

Ejemplos/Clase: 500	
Exactitud	0.569
Precisión	0.585
Recall	0.569
F1	0.571
MCC	0.522

Tabla A.86: Resultados de la Red Neuronal Profunda con Normalización por Lotes sobre CIFAR-10.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.22.

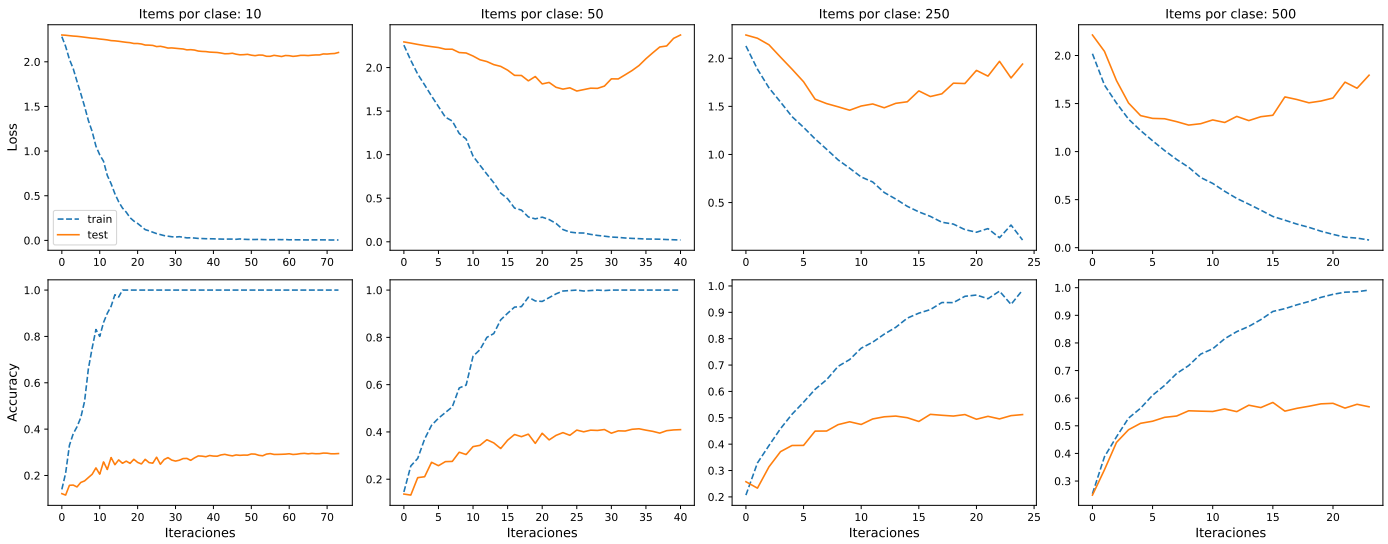

Figura A.22: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Normalización por Lotes sobre CIFAR-10.

A.3.8. Red Neuronal Profunda con función de costo Similitud de Coseno

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.87.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	20	18	17	17
Tamaño Lote	20	28	32	48
Tasa de Aprendizaje	0.00005			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.87: Tabla de parámetros para Red Neuronal Profunda con función de costo Similitud de Coseno sobre CIFAR-10.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.88.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	32x32x3				
Convolucional		(7,7)	128		ReLU
AgrupaciónMáxima		(4,4)			
Convolucional		(7,7)	128		ReLU
Convolucional		(7,7)	64		ReLU
Convolucional		(7,7)	64		ReLU
Aplanar					
Densa				48	ReLU
Densa				32	ReLU
Densa				10	Softmax

Tabla A.88: Tabla resumen de la Estructura de Red Neuronal Profunda con función de costo Similitud de Coseno sobre CIFAR-10.

Los resultados obtenidos por la Red Neuronal Profunda con función de costo Similitud de Coseno se presentan en la Tabla A.89.

Ejemplos/Clase: 10	
Exactitud	0.269
Precisión	0.288
Recall	0.269
F1	0.262
MCC	0.191

Ejemplos/Clase: 50	
Exactitud	0.375
Precisión	0.395
Recall	0.375
F1	0.373
MCC	0.307

Ejemplos/Clase: 250	
Exactitud	0.508
Precisión	0.503
Recall	0.508
F1	0.504
MCC	0.454

Ejemplos/Clase: 500	
Exactitud	0.573
Precisión	0.58
Recall	0.573
F1	0.569
MCC	0.528

Tabla A.89: Resultados de la Red Neuronal Profunda con función de costo Similitud de Coseno sobre CIFAR-10.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.23.

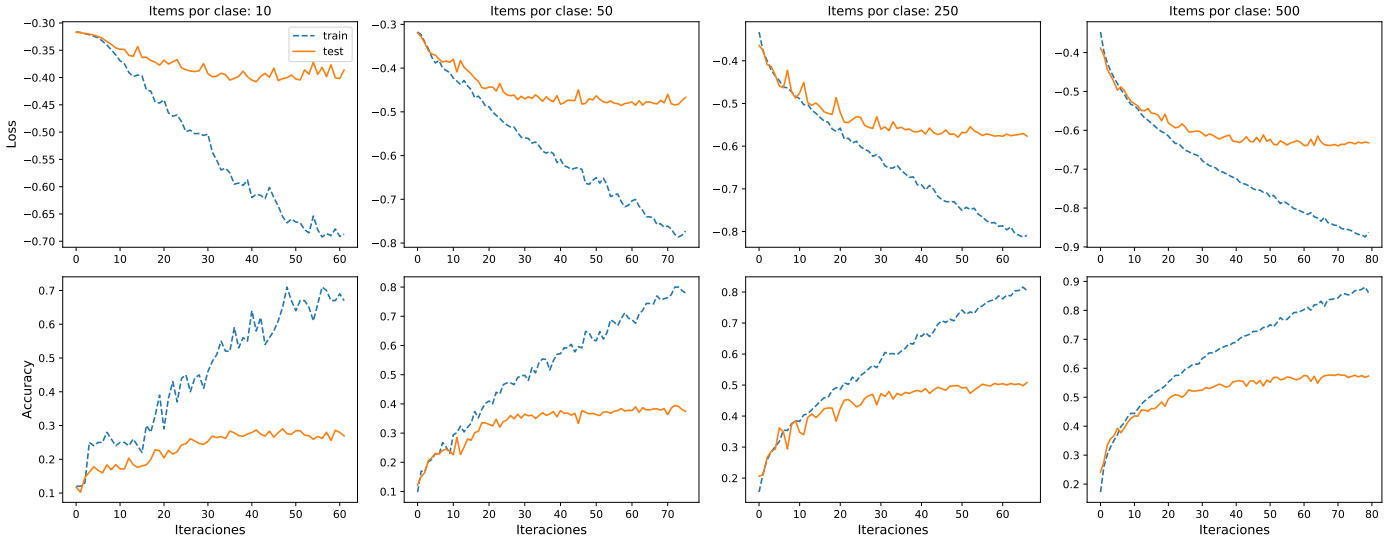

Figura A.23: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con función de costo Similitud de Coseno sobre CIFAR-10.

A.3.9. Red Neuronal Profunda con Convolución Dilatada

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.90.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	30	15	10	10
Tamaño Lote	20	32		
Tasa de Aprendizaje	0.00005			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.90: Tabla de parámetros para Red Neuronal Profunda con Convolución Dilatada sobre CIFAR-10.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.91.

Capa	Tamaño	Kernel	Filtros	Tasa de Dilatación	Unidades	Activación
Entrada	32x32x3					
Convolucional		(7,7)	128	2		ReLU
AgrupaciónMáxima		(4,4)				
Convolucional		(7,7)	128	2		ReLU
Convolucional		(7,7)	64	2		ReLU
Convolucional		(7,7)	64	2		ReLU
Aplanar						
Densa					48	ReLU
Densa					32	ReLU
Densa					10	Softmax

Tabla A.91: Tabla resumen de la Estructura de Red Neuronal Profunda con Convolución Dilatada sobre CIFAR-10.

Los resultados obtenidos por la Red Neuronal Profunda con Convolución Dilatada se presentan en la Tabla A.92.

Ejemplos/Clase: 10	
Exactitud	0.262
Precisión	0.272
Recall	0.262
F1	0.254
MCC	0.184

Ejemplos/Clase: 50	
Exactitud	0.350
Precisión	0.353
Recall	0.35
F1	0.341
MCC	0.28

Ejemplos/Clase: 250	
Exactitud	0.473
Precisión	0.478
Recall	0.473
F1	0.469
MCC	0.415

Ejemplos/Clase: 500	
Exactitud	0.531
Precisión	0.533
Recall	0.531
F1	0.524
MCC	0.48

Tabla A.92: Resultados de la Red Neuronal Profunda con Convolución Dilatada sobre CIFAR-10.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.24.

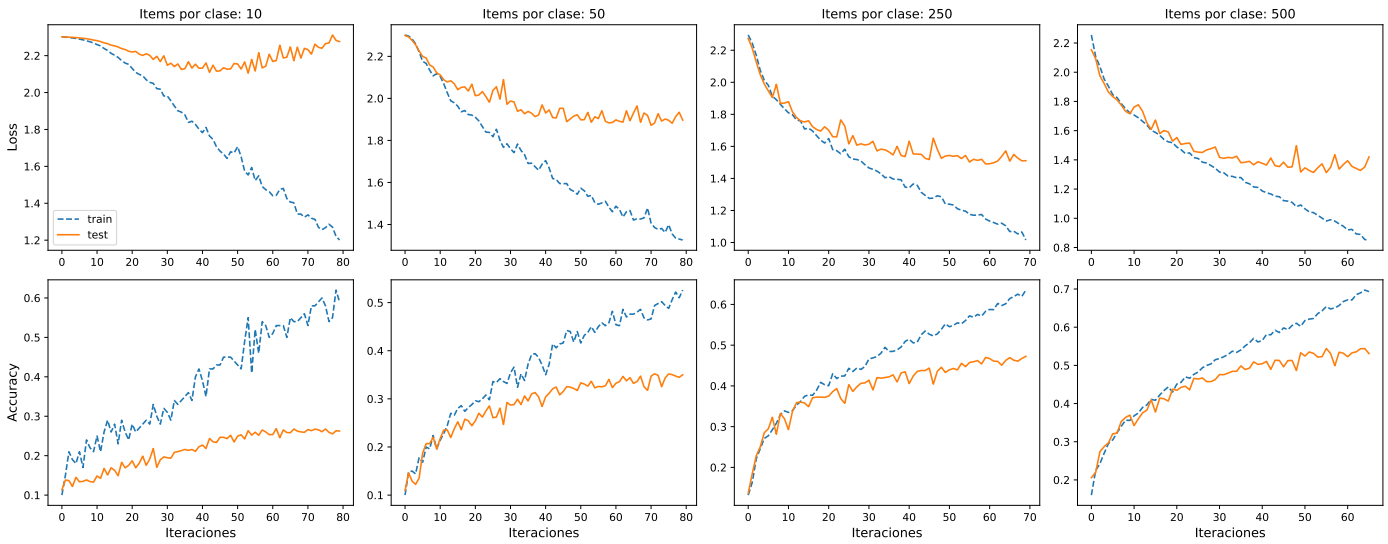

Figura A.24: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Convolución Dilatada sobre CIFAR-10.

A.3.10. Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.93.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	65	48	27	17
Tamaño Lote	20	32		
Tasa de Aprendizaje mínima	0.00005			
Tasa de Aprendizaje máxima	0.00015			
Delta (Decadencia cíclica de tasa de aprendizaje)	10			
s (Decadencia cíclica de tasa de aprendizaje)	20			
Limite de iteraciones (Entrenamiento)	80			

Tabla A.93: Tabla de parámetros para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre CIFAR-10.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la

Tabla A.94.

Capa	Tamaño	Kernel	Filtros	Unidades	Activación
Entrada	32x32x3				
Convolucional		(7,7)	128		ReLU
AgrupaciónMáxima		(4,4)			
Convolucional		(7,7)	128		ReLU
Convolucional		(7,7)	64		ReLU
Convolucional		(7,7)	64		ReLU
Aplanar					
Densa				48	ReLU
Densa				32	ReLU
Densa				10	Softmax

Tabla A.94: Tabla resumen de la Estructura de Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre CIFAR-10.

Los resultados obtenidos por la Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje se presentan en la Tabla A.95.

Ejemplos/Clase: 10	
Exactitud	0.288
Precisión	0.282
Recall	0.288
F1	0.270
MCC	0.212

Ejemplos/Clase: 50	
Exactitud	0.360
Precisión	0.353
Recall	0.36
F1	0.349
MCC	0.291

Ejemplos/Clase: 250	
Exactitud	0.507
Precisión	0.498
Recall	0.507
F1	0.500
MCC	0.453

Ejemplos/Clase: 500	
Exactitud	0.562
Precisión	0.563
Recall	0.562
F1	0.559
MCC	0.514

Tabla A.95: Resultados de la Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre CIFAR-10.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.25.

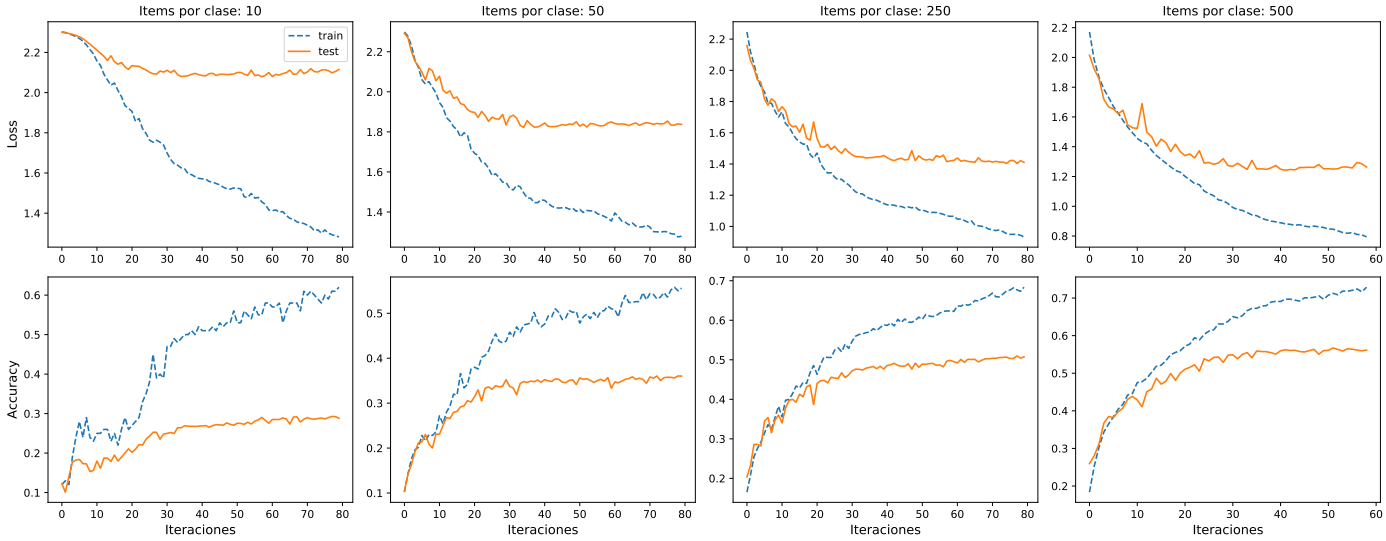

Figura A.25: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con Decadencia Cíclica de Tasa de Aprendizaje sobre CIFAR-10.

A.3.11. Red Neuronal Profunda con combinación de técnicas (C1)

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.96.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	90	35	30	20
Tamaño Lote	20	32		
Tasa de Aprendizaje mínima	0.0005			
Tasa de Aprendizaje máxima	0.001			
Delta (Decadencia cíclica de tasa de aprendizaje)	10			
s (Decadencia cíclica de tasa de aprendizaje)	10			
Limite de iteraciones (Entrenamiento)	350			

Tabla A.96: Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas sobre CIFAR-10.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.97.

Capa	Tamaño	Kernel	Filtros	Tasa de Dilatación	Probabilidad	Unidades	Activación
Entrada	28x28x1						
Convolucional		(7,7)	128	2			ReLU
NormalizaciónPorLotes							
AgrupaciónMáxima		(4,4)					
Convolucional		(7,7)	128	2			ReLU
Convolucional		(7,7)	64	2			ReLU
Convolucional		(7,7)	64	1			ReLU
AgrupaciónPromedioGlobal							
Dropout					0.2		
Densa						10	Softmax

Tabla A.97: Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas sobre CIFAR-10.

Los resultados obtenidos por la Red Neuronal Profunda con combinación de técnicas se presentan en la Tabla A.98.

Ejemplos/Clase: 10	
Exactitud	0.291
Precisión	0.302
Recall	0.291
F1	0.293
MCC	0.212

Ejemplos/Clase: 50	
Exactitud	0.385
Precisión	0.394
Recall	0.385
F1	0.386
MCC	0.317

Ejemplos/Clase: 250	
Exactitud	0.469
Precisión	0.475
Recall	0.469
F1	0.466
MCC	0.412

Ejemplos/Clase: 500	
Exactitud	0.544
Precisión	0.541
Recall	0.544
F1	0.540
MCC	0.494

Tabla A.98: Resultados de la Red Neuronal Profunda con combinación de técnicas sobre CIFAR-10.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.26.

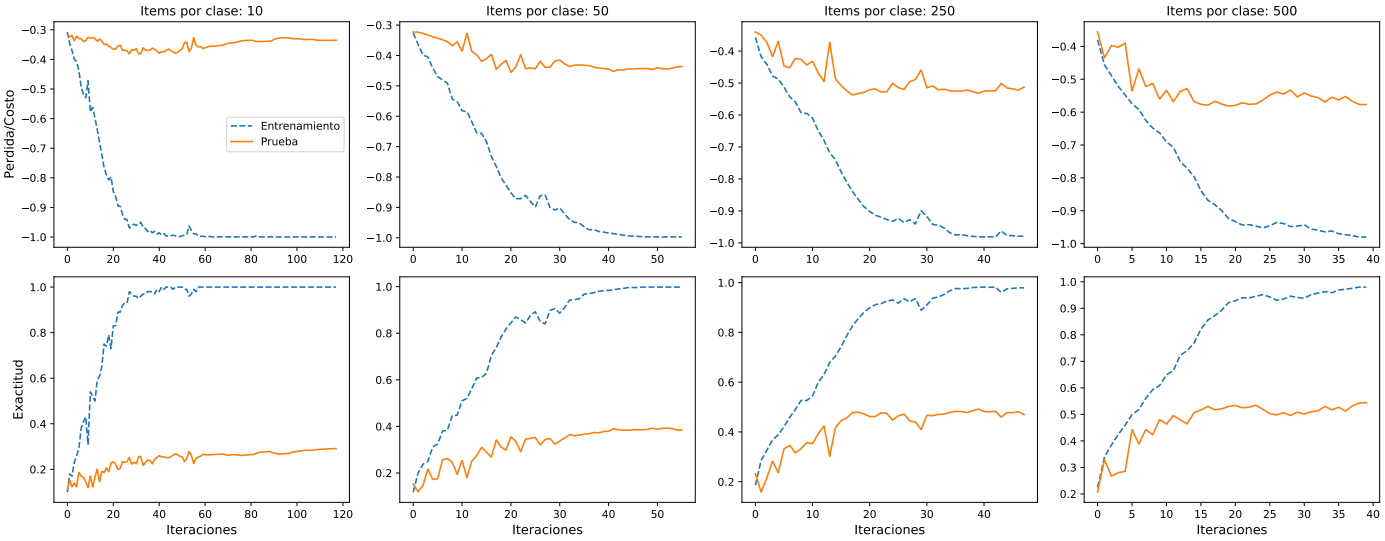

Figura A.26: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas sobre CIFAR-10.

A.3.12. Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento artificial de datos (C2)

Los parámetros utilizados para la implementación de la Red Neuronal Profunda se presentan en la Tabla A.99.

	Ejemplos por clase			
Parámetros	10	50	250	500
Paciencia (Detención temprana)	25	25	25	20
Tamaño Lote	20	32		
Interpolación	Vecino más cercano			
Rango de rotación (Aumento de datos)	0º a 45º			
Desplazamiento horizontal (Aumento de datos)	10 %			
Desplazamiento vertical (Aumento de datos)	10 %			
Rango de acercamiento (Aumento de datos)	85 % a 100 %			
Volteo horizontal (Aumento de datos)	Si			
Tasa de Aprendizaje mínima	0.00005			
Tasa de Aprendizaje máxima	0.00015			
Delta (Decadencia cíclica de tasa de aprendizaje)	10			
s (Decadencia cíclica de tasa de aprendizaje)	10			
Limite de iteraciones (Entrenamiento)	100			

Tabla A.99: Tabla de parámetros para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento de la cantidad de datos sobre CIFAR-10.

Las capas que componen la red neuronal, junto con sus parámetros se puede observar en la Tabla A.100.

Capa	Tamaño	Tasa de Aumento	Probabilidad	Unidades	Activación
Entrada	32x32x3				
Ampliar		(7,7)			
EfficientNetB0					
AgrupaciónPromedioGlobal					
NormalizaciónPorLotes					
Dropout			0.25		
Densa				10	Softmax

Tabla A.100: Tabla resumen de la Estructura de Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento de la cantidad de datos sobre CIFAR-10.

Los resultados obtenidos por la Red Neuronal Profunda con combinación de técnicas más

Transferencia de aprendizaje y Aumento artificial de datos se presentan en la Tabla A.101.

Ejemplos/Clase: 10	
Exactitud	0.378
Precisión	0.39
Recall	0.378
F1	0.351
MCC	0.315

Ejemplos/Clase: 50	
Exactitud	0.584
Precisión	0.617
Recall	0.584
F1	0.574
MCC	0.544

Ejemplos/Clase: 250	
Exactitud	0.724
Precisión	0.728
Recall	0.724
F1	0.717
MCC	0.695

Ejemplos/Clase: 500	
Exactitud	0.762
Precisión	0.768
Recall	0.762
F1	0.758
MCC	0.738

Tabla A.101: Resultados de la Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento de la cantidad de datos sobre CIFAR-10.

La evolución del costo y de la exactitud en los conjuntos de entrenamiento y prueba durante el proceso de entrenamiento se presentan en la Figura A.27.

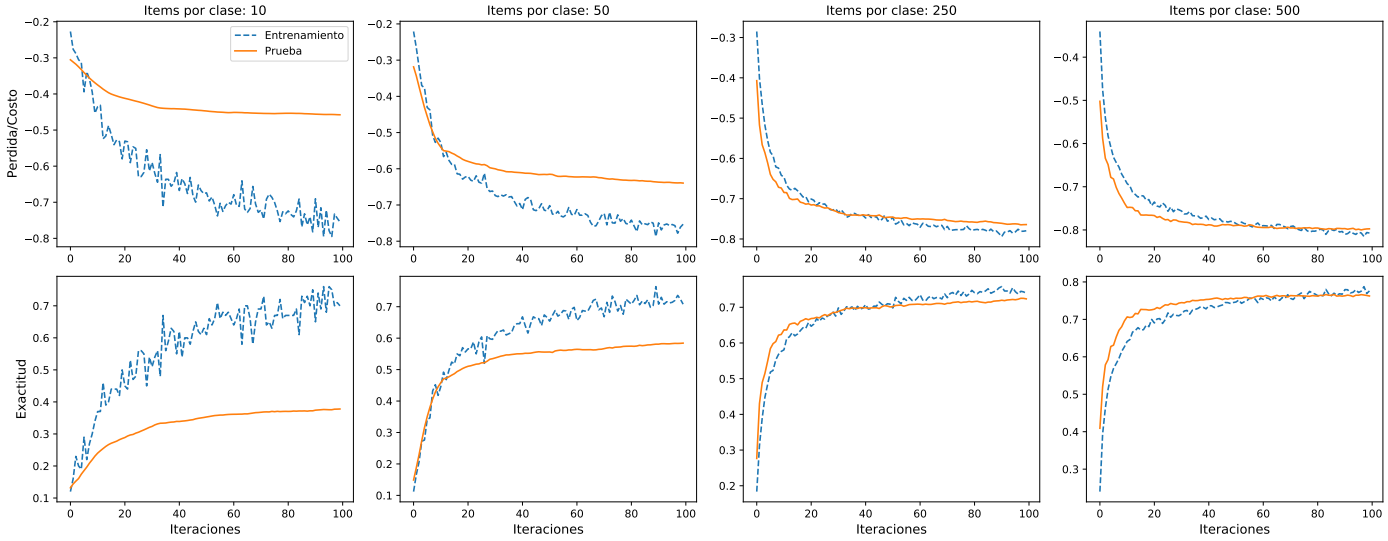

Figura A.27: Gráficas de costo y Exactitud durante entrenamiento para Red Neuronal Profunda con combinación de técnicas más Transferencia de aprendizaje y Aumento de la cantidad de datos sobre CIFAR-10.

A.3.13. Ensamblaje de todas las Redes Neuronales Profundas

Los resultados obtenidos por el ensamblaje de redes neuronales se presentan en la Tabla A.102.

Ejemplos/Clase: 10	
Exactitud	0.319
Precisión	0.315
Recall	0.319
F1	0.310
MCC	0.245

Ejemplos/Clase: 50	
Exactitud	0.430
Precisión	0.432
Recall	0.430
F1	0.426
MCC	0.368

Ejemplos/Clase: 250	
Exactitud	0.567
Precisión	0.558
Recall	0.567
F1	0.560
MCC	0.519

Ejemplos/Clase: 500	
Exactitud	0.642
Precisión	0.637
Recall	0.642
F1	0.639
MCC	0.603

Tabla A.102: Resultados del ensamblaje de todas las Redes Neuronales Profundas sobre CIFAR-10.