Estudio de clasificador *feed-forward* pre-entrenado con un *autoencoder* convolucional

Redes Neuronales - Trabajo Final

Franco Nicolás Nieto - franconicolasnieto@gmail.com

Estudiante de grado de Cs. Físicas en la Facultad de Ciencias Exactas y Naturales (FCEN) - UBA

Segundo Cuatrimestre 2023, Cátedra Francisco A. Tamarit

Facultad de Matemática, Astronomía, Física y Computación (FAMAF) - UNC

Resumen

En el presente trabajo se estudia una red neuronal clasificadora *feed-forward* pre-entrenada con un *autoencoder* convolucional. En primer lugar se entrenaron dos *autoencoders*, uno de referencia y otro que a partir de disminuir la probabilidad de *Dropout* p y aumentar el número de neuronas se logró reducir el *MSE*. Luego dos redes clasificadoras heredaron los *encoders* de estos *autoencoders* y se observó que el desempeño de la red con menor p dependía de las condiciones iniciales del clasificador debido a su reducida estocasticidad que se solucionó al aumentar nuevamente p. Posteriormente se caracterizó estadísticamente la red con menos neuronas y se calculó su matriz de confusión. Por último, entrenando un clasificador idéntico al anterior pero con los parámetros del *encoder* generados aleatoriamente se observó el mismo desempeño que el clasificador pre-entrenado indicando que para esta red simple no resulta relevante el pre-entrenamiento.

1. Introducción

1.1. Motivación

Las redes neuronales convolucionales son una herramienta fundamental para el procesamiento de imágenes que hace posible la visión por computadora. Este tipo de red conserva la topología de la entrada y logra aprovechar la información contenida en la distribución espacial de los datos como por ejemplo, la distribución de los píxeles en una imagen. Adicionalmente, para entrenar un clasificador de red profunda se precisa de un pre-entrenamiento debido a la supresión del gradiente que se da en redes de este estilo. Es por estas razones que resulta interesante estudiar estas dos herramientas, las capas convolucionales y el pre-entrenamiento.

1.2. Conceptos básicos

Las redes neuronales *feed-forward* tienen la particularidad de que la información fluye desde la entrada hacia la salida pasando por cada una de las capas en una sola dirección sin retroalimentación. Por otra parte, una red *autoencoder* es una arquitectura que se entrena para poder representar los datos de entrada de una manera eficiente reduciendo su dimensionalidad y luego decodificando esta información. En su entrenamiento el objetivo es lograr una función similar a la identidad.

Por último, las redes convolucionales emplean filtros convolucionales y de *pooling* y mapean la imagen de entrada en capas de diferente tamaño (ancho, alto y profundidad), pero conservan la misma topología. Estos filtros permiten conservar la información codificada dentro de la distribución espacial de los píxeles de la entrada que es clave para la visión por computadora ya que allí es donde yace la esencia de una imagen y esta pierde el sentido si se desordenaran los píxeles. En este caso el entrenamiento se llevará a cabo con imágenes etiquetadas de la base de datos *Fashion-MNIST* [2].

1.3. Funciones de error y optimización

Las redes neuronales que se utilizaron en este trabajo emplearon el algoritmo de optimización *ADAM*(*Adaptative Moment Estimation*) [1] ya que es un método adaptativo con memoria que varía el *learning rate* automáticamente según las actualizaciones previas de pesos sinápticos.

En cuanto a las funciones de error utilizadas se tuvieron que seleccionar dos. Una fue el Error Cuadrático Medio (*MSE*) para validar el *autoencoder*, esta está dada por la ecuación 1 donde J_i^μ y O_i^μ son el valor real y la predicción del i-ésimo píxel de la μ-ésima imagen de entrada, N es el número total de píxeles en la imagen y P es el número total de imágenes en el *minibatch*

$$E[\bar{\omega}] = \frac{1}{2}\sum_{\mu=1}^{P}\sum_{i=1}^{N}[J_i^\mu - O_i^\mu]^2 \tag{1}$$

La otra función utilizada fue el Error de Entropía

Cruzada (*CEL*) para el clasificador ya que devuelve Funciones de Masa de Probabilidad (*PMF*) o distribuciones discretas. Este error está dado por la Ecuación 2, donde $Q(x)$ es la probabilidad predicha por la red de que la entrada pertenezca a la categoría x, y $P(x)$ es la probabilidad real. Como en este caso $P(x = \ell) = 1$ y $P(x \neq \ell) = 0$ donde ℓ es la etiqueta correcta de la imagen de entrada la función *CEL* queda como el término de la derecha.

$$H(P,Q) = -\sum_{x}^{N} P(x) log_2[Q(x)] = -log_2[Q(\ell)] \quad (2)$$

1.4. Estructura y capas de la red

Los datos con los que se entrenará y validará la red neuronal provienen de la base de datos *Fashion-MNIST* [2] que consiste en un *dataset* de 60.000 imágenes de entrenamiento y 10.000 de prueba. Estas son imágenes de 10 categorías de ropa y tienen un tamaño de 28×28 píxeles en escala de grises que toman valores entre 0 y 255.

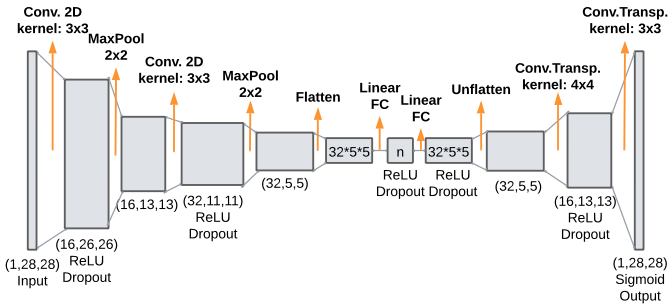

Figura 1: Esquema de la red neuronal *feed-forward autoencoder* convolucional utilizada para el pre-entrenamiento de la red clasificadora. Cada rectángulo representa cada capa con sus respectivas dimensiones y funciones que se le aplican previo a introducirla dentro de la siguiente transformación indicada con una flecha naranja.

La red *autoencoder* convolucional con la que se procesará el pre-entrenamiento del clasificador se encuentra esquematizada en la Figura 1 con sus respectivas transformaciones y dimensiones de capas con ReLUs y Dropouts de probabilidad p. Esta consistió en un *encoder* cuya entrada es una imagen de 28×28 en escala de grises del *dataset* Fashion-MNIST que luego de dos capas de convolución con su respectivo MaxPooling se mapeó con un *flatten* en una capa lineal de $32 * 5 * 5$ neuronas totalmente conectado con otra de n neuronas para su posterior decodificación. Esta etapa llamada *decoder* empieza conectando totalmente la capa lineal de tamaño n con una de $32*5*5$ neuronas que luego através de un *unflatten* y dos convoluciones transpuestas (con *stride: 2* y *output_padding: 1*) se mapeó nuevamente en una imagen de 28×28 en escala de grises.

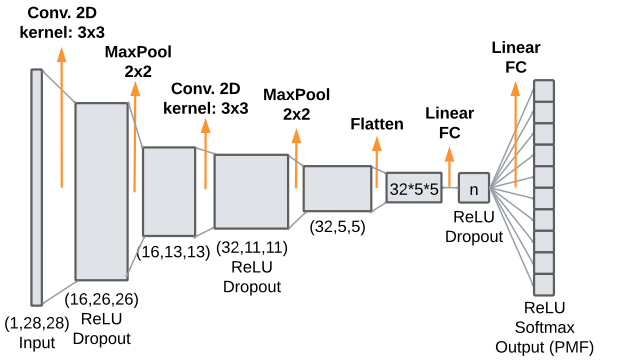

Figura 2: Esquema de la red neuronal clasificadora *feed-forward* convolucional. Cada rectángulo representa cada capa con sus respectivas dimensiones y funciones que se le aplican previo a introducirla dentro de la siguiente transformación indicada con una flecha naranja. La salida consiste en una capa lineal de 10 neuronas e indican la probabilidad predicha para que el *input* pertenezca a su categoría de ropa asociada.

En segundo lugar, la estructura de la red clasificadora se observa en el esquema de la figura 2. Esta hereda del *autoencoder* la etapa de *encoder* junto con sus pesos sinápticos como pre-entrenamiento. Posteriormente la capa lineal de n neuronas está totalmente conectada con otra capa lineal de 10 neuronas. Esta salida, luego de una ReLU y una SoftMax, es la distribución de probabilidad predicha de que la entrada pertenezca a cada una de las 10 categrías de ropa.

2. Desarrollo y Resultados

2.1. Pre-entrenamiento con autoencoder

Una vez construido el modelo de *autoencoder* mencionado previamente comienza la etapa de entrenamiento de la red. El objetivo en este apartado es encontrar la mejor combinación de hiperparámetros que minimicen el error (*MSE*), para esto compararemos con un modelo base de hiperparámetros *(n=64, p=0,2, learning rate=* 10^{-3}*, tamaño de minibatch= 100)* y a partir de este se buscará mejorar. El entrenamiento de este modelo base se puede apreciar en la curva azul y roja de la Figura 3, se puede observar que a partir de las 10 épocas su decrecimiento se frena y para la época 20 alcanza finalmente un $MSE = 0,037$.

Con el objetivo de minimizar este *MSE* se mantuvieron todos los hiperparámetros fijos excepto uno que se fue variando. Al variar tanto el *learning rate* como el tamaño del *minibatch* en ambas direcciones se observó que el *MSE* crecía. Sin embargo el estudio de la cantidad de neuronas n (Tabla 1) y la probabilidad de *Dropout* p (Tabla 2) resultó más interesante.

Como se puede observar a partir de estas tablas, el valor de p que minimiza el *MSE* es $p = 0,01$ que se

n	64	128	256	512
MSE	0,037	0,031	0,031	0,029

Tabla 1: *MSE* para redes de diferente n, con $p = 0,2$.

p	0,2	0,05	0,01	0,0001
MSE	0,037	0,025	0,023	0,023

Tabla 2: *MSE* para redes de diferente p, con $n = 64$.

elegirá por sobre $p = 0,001$ para mantener una mayor estocasticidad. Por otra parte, notamos que a medida que aumenta n, y con ello el tiempo de cómputo, disminuye *MSE*. Por esta razón el modelo que se elegirá para contrastar con el base justamente utilizará los hiperparámetros *(n=128, p=0,01, learning rate=* 10^{-3}*, tamaño de minibatch= 100)*. Su entrenamiento se puede observar junto con el del modelo base en la Figura3.

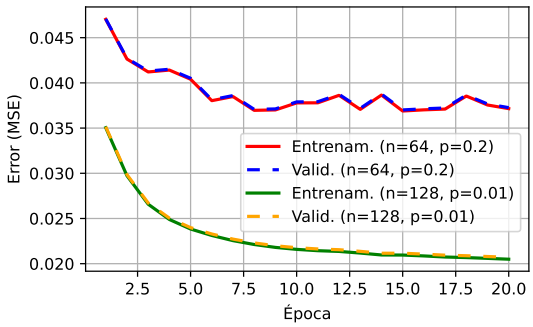

Figura 3: Curvas de *MSE* sobre los conjuntos de entrenamiento y de validación durante el entrenamiento de los dos modelos de *autoencoders* utilizados. Uno base con $n = 64$ y $p = 0,2$ y otro con $n = 128$ y $p = 0,01$.

Se hace evidente que el nuevo *MSE* es menor que el del modelo base y a su vez se suaviza la curva debido a que la reducción del p disminuye la estocasticidad y el modelo se mantiene dentro de un mismo pozo local. Como resultado se obtuvo un *autoencoder* con $MSE = 0,021$ reduciendo considerablemente lo obtenido para el modelo base. En el apéndice se compara la capacidad de filtrado de ambos modelos.

2.2. Comparación de clasificadores

Los dos modelos de clasificador que se utilizaron heredaron los hiperparámetros de los *autoencoders* así como los pesos de sus respectivos *encoders*, el base con (*n=64 y p=0,2*) y el mejorado con (*n=128 y p=0,01*) . Una vez pre-entrenados se procedió con su entrenamiento final que se observa en la Figura 4. Se nota que el desempeño del modelo mejorado (curva azul) está sujeto a las condiciones con las que se inicializa la capa del clasificador, esto se observa al ver que no es capaz de mejorar considerablemente más allá de su punto de partida.

Por esta razón y para lograr reducir el sobre-ajuste,

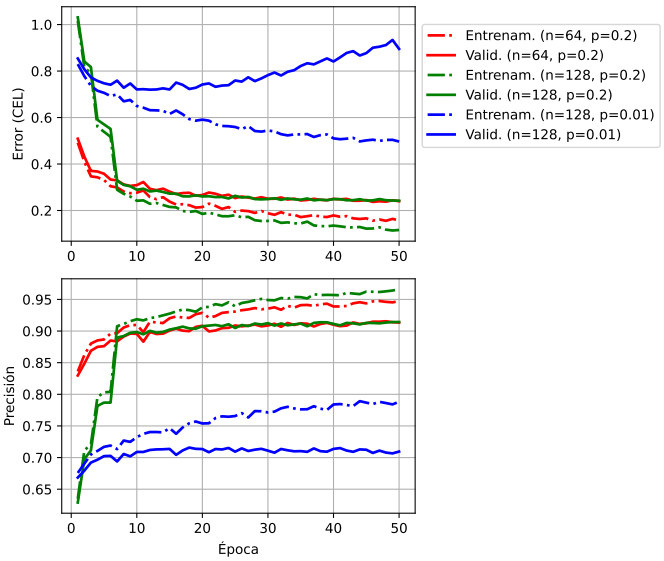

Figura 4: Curvas de *CEL* y precisión sobre los conjuntos de entrenamiento y de validación durante el entrenamiento de las tres variantes de clasificador. Uno base (rojo) con $n = 64$ y $p = 0,2$ y otros dos con $n = 128$ de $p = 0,01$ (azul) y $p = 0,2$ (verde).

se utilizó en la etapa de entrenamiento del clasificador de $n = 128$ una probabilidad de *Dropout* de $p = 0,2$ para brindarle una mayor estocasticidad al modelo. Cabe aclarar que el pre-entrenamiento sigue siendo el mismo, es decir, el proveniente de un *autoencoder* con (*n=128 y p=0,01*), solo se alteró p luego de heredar los parámetros. Habiendo introducido esta variación se apreció una mejora considerable que se puede observar en la curva verde de la Figura 4. Esta, a pesar de tener unas condiciones iniciales desfavorables, rápidamente logra aumentar la precisión y acercarse a las curvas del modelo base. También es destacable el hecho de que el aumento de p permita al entrenamiento hacer los saltos que se observan alrededor de la época 5.

Sin embargo, queda resaltar un último aspecto de estos gráficos, a pesar de la alteración introducida al modelo de $n = 128$ no se logró reducir lo suficiente el sobre-ajuste para igualar el del modelo de $n = 64$. Por lo tanto se concluye que los modelos pre-entrenados con el *autoencoder* de $n = 128$ y $p = 0,01$ que tanto reducía el *MSE* no resultaron superadores al modelo base como para justificar el incremento de tiempo de cómputo. Es por esto que se procederá a caracterizar el modelo base con mayor profundidad.

2.3. Caracterización clasificador

El objetivo de esta sección es mejorar la caracterización del clasificador base de $n = 64$ y $p = 0,2$. Con el objetivo de caracterizar las fluctuaciones estadísticas del entrenamiento se entrenaron 15 redes compuestas por mismo modelo, utilizando para to-

das el pre-entrenamiento generado por el *autoencoder* base con $MSE = 0,037$ expuesto previamente. Así se obtuvieron diversas curvas de precisión que se promediaron y se les calculó el desvío estándar para cada época obteniendo la Figura 5. Se observó que para 30 épocas la precisión es de $(91,1 \pm 0,3)\,\%$ y para la época 50 es de $(91,4 \pm 0,2)\,\%$. Por lo tanto, como no hay diferencias significativas entre ambos valores y el entrenar a la red solamente hasta la época 30 reduce el tiempo de cómputo en un 40 % se considera que, si bien no se llega al punto óptimo de precisión, es una cantidad de épocas apropiada si se tiene en cuenta la relación precisión-tiempo.

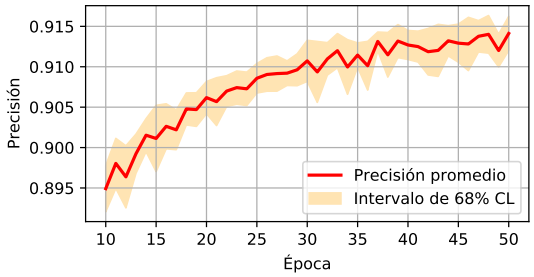

Figura 5: Análisis estadístico de la curva de precisión sobre el conjunto de validación utilizando 15 modelos de $n = 64$ y $p = 0,2$ pre-entrenados con el mismo *autoencoder* de $MSE = 0,037$.

En la figura 6 se presenta la matriz de confusión resultante de este modelo base de 30 épocas utilizando el conjunto de validación. Notar que estos números podrían mejorarse a cambio de un mayor tiempo de cómputo ya que continúa la tendencia ascendente en la curva de precisión.

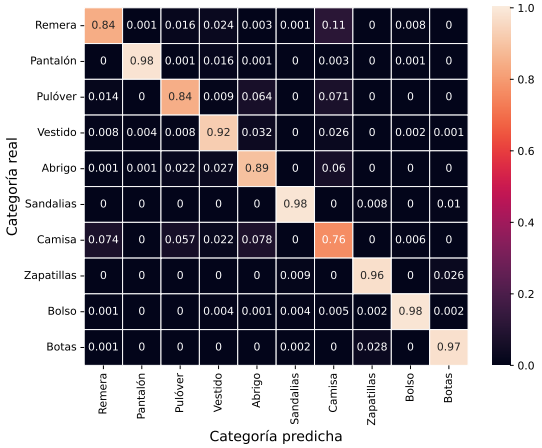

Figura 6: Matriz de confusión del clasificador pre-entrenado con $n = 64$ y $p = 0,2$.

2.4. Impacto del pre-entrenamiento

Para analizar si efectivamente el pre-entrenamiento con el *autoencoder* resultó ventajoso se entrenó una red con la misma estructura que en la sección anterior pero inicializando los parámetros del encoder de forma aleatoria, es decir sin pre-entrenamiento. La comparación con el modelo entrenado se puede observar en la Figura 7.

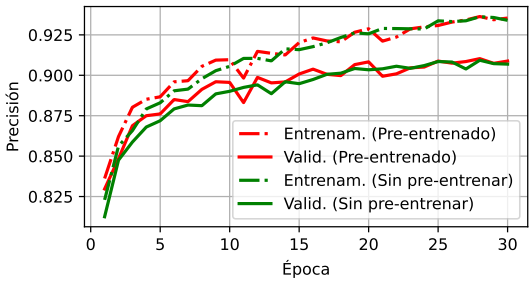

Figura 7: Curvas de precisión sobre los conjuntos de entrenamiento y de validación durante el entrenamiento de dos clasificadores con misma estructura $n = 64$ y $p = 0,2$, una pre-entrenada (rojo) y otra no (verde).

Como se puede observar, ambas curvas son muy similares, si bien el modelo no pre-entrenado comienza con una precisión menor que el pre-entrenado, rápidamente lo alcanza y terminan con una precisión idéntica. Esto nos da el indicio de que, al menos para un modelo tan simple como el discutido en este trabajo, no se logra aprovechar el pre-entrenamiento que suele ser más útil en redes con mayor profundidad.

3. Conclusiones

En este trabajo se logró crear un *autoencoder* convolucional que filtre imágenes con un $MSE = 0,037$ y a partir del estudio de sus hiperparámtros se mejoró el modelo reduciendo la probabilidad de *Dropout* p y aumentando el número de neuronas. Así este nuevo *autoencoder* consiguió un $MSE = 0,021$.

Posteriormente se utilizaron los *encoders* de ambos modelos para pre-entrenar dos redes clasificadoras. Se observó que el desempeño de la red con menor *Dropout* era sumamente dependiente de las condiciones iniciales de la etapa de clasificación y no lograba mejorar considerablemente, esto se atribuyó a la reducción de estocasticidad. Al aumentar nuevamente p se solucionó este aspecto pero no sobrepasó el desempeño del clasificador con menos neuronas. Por esto se procedió a caracterizar este último. Se entrenó 15 veces esta misma red con los mismos parámetros de pre-entrenamiento para conocer la fluctuación estadística de la precisión y se confeccionó la matriz de confusión para esta red entrenada con 30 épocas.

Por último se puso a prueba la importancia del pre-entrenamiento entrenando una red idéntica a la anterior pero inicializando el *encoder* con parámetros aleatorios y sorpresivamente se observó que el desempeño de ambas fue idéntico. Esto indica que para esta estructura simple de red no resulta relevante hacer un pre-entrenamiento.

Referencias

[1] Cornell University, *Adam*, 2003, https://optimization.cbe.cornell.edu/index.php?title=Adam

[2] Fashion-MNIST, 17/2/2024. https://github.com/zalandoresearch/fashion-mnist

Apéndice

Comparación de *autoencoders*

Como se comentó en la sección 2.1, a lo largo del trabajo se compararon dos modelos con diferentes pre-entrenamientos, el modelo base surge de un *autoencoder* con hiperparámetros *(n=64, p=0,2, learning rate*= 10^{-3}*, tamaño de minibatch= 100)* y $MSE = 0,037$ y otro modelo proviene de una mejora con un $MSE = 0,021$ a partir de la elección de $n = 128$ y $p = 0,01$. En la figura 8 se compara la capacidad de filtrado de ambos modelos.

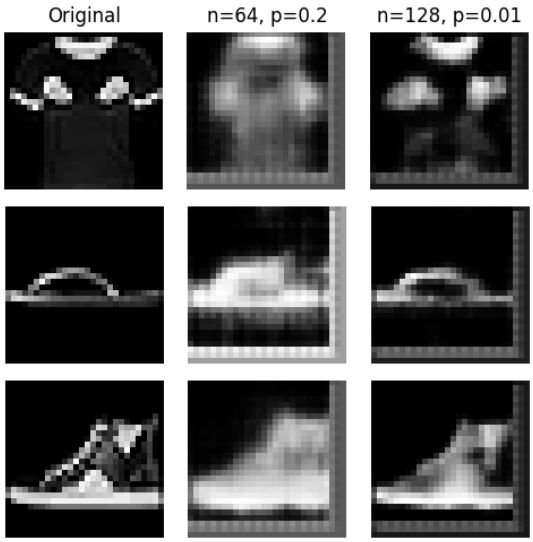

Figura 8: Comparación de filtrado de imágenes de los dos modelos de *autoencoders* utilizados. Uno base con $n = 64$ y $p = 0,2$ y otro con $n = 128$ y $p = 0,01$.

Se puede observar que el modelo mejorado produce imágenes más nítidas como para la ojota y con mayor detalle como en el caso de la remera y la zapatilla. Si bien esto conduce a un menor *MSE* para el *autoencoder*, también puede producir que las imágenes tengan demasiado detalle para el clasificador posterior y no ayuden a destacar el patrón general de fondo. Notar que para la remera, la red de $n = 64$ logra producir la silueta de una remera genérica que puede ayudar al clasificador mientras que la red de $n = 128$ mantiene las zonas negras de la imagen.