Autoencoders

- Banfi, Malena
- Fleischer, Lucas
- Perez Rivera, Mateo
- Szejer, Ian

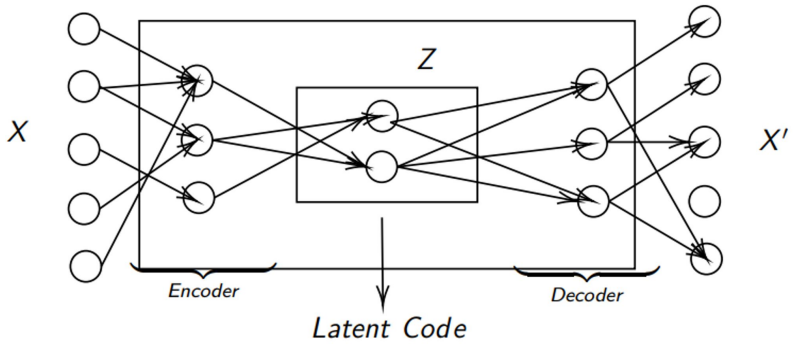
Contenido

1. Autoencoder
2. Denoising Autoencoder
3. Variational Autoencoder

Autoencoder

Autoencoder

- Son la base de algunos modelos de redes neuronales generativas.
- Está conformado por dos redes neuronales de perceptrones multicapa, donde la salida de la primera red se conecta con la entrada de la segunda red, la cual tiene la distribución invertida de neuronas en las capas y como salida tiene la misma dimensión que la entrada de la primera red.
- El objetivo de la red es la optimización, se busca minimizar el error en la salida de la segunda red.
- La información queda representada en la capa latente.



Ejercicio 1.a

Implementar un Autoencoder básico para las imágenes binarias de la lista de caracteres del archivo ”font.h”.

1. Plantear una arquitectura de red para el Codificador y Decodificar que permita representar los datos de entrada en un espacio latente de dos dimensiones.
2. Describan y estudien las diferentes arquitecturas y técnicas de optimización que fueron aplicando para permitir que la red aprenda todo el set de datos o un subconjunto del mismo. El objetivo es aprender los 32 patrones de 5*7 en un espacio latente de 2 dimensiones con un error máximo de 1 pixel incorrecto. En el caso de que sea un subconjunto mostrar porque no fue posible aprender el dataset completo.
3. Realizar el gráfico en dos dimensiones que muestre los datos de entrada en el espacio latente.
4. Mostrar cómo la red puede generar una nueva letra que no pertenece al conjunto de entrenamiento.

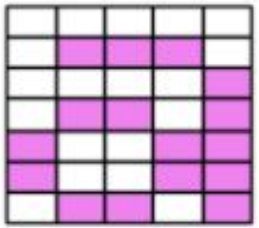
Utilización de Jupyter

```
{0x00, 0x0e, 0x01, 0x0d, 0x13, 0x13, 0x0d},   // 0x61, a
```

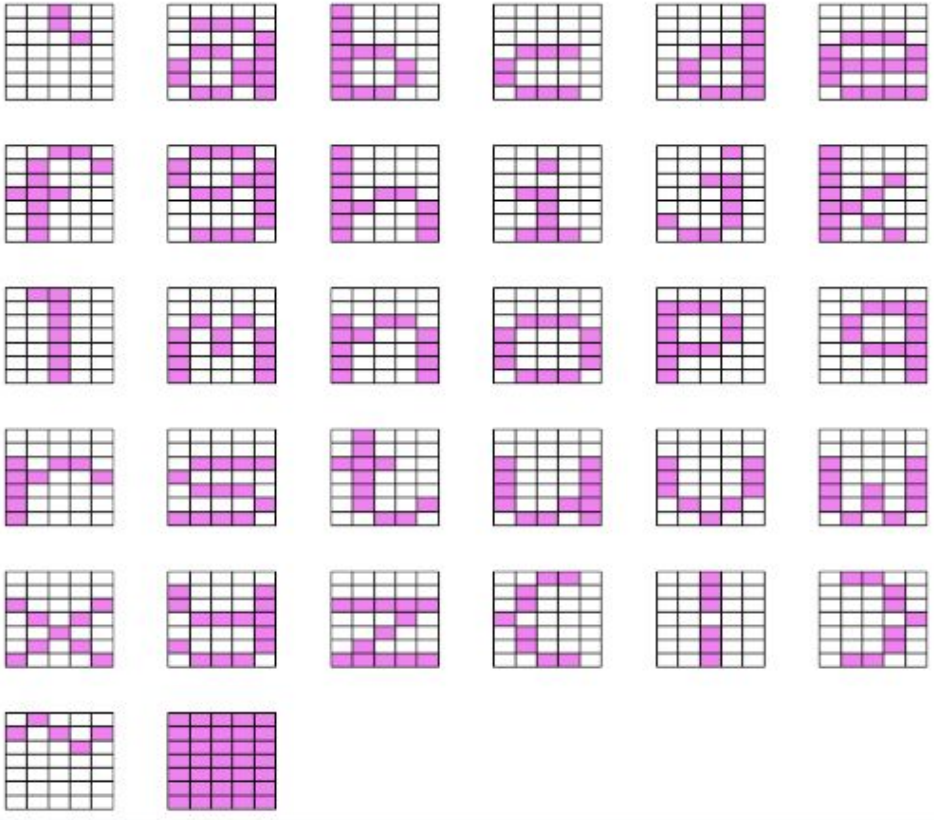
```
char_example = np.copy(font[1])
char_example.resize(7, 1)
print(char_example)
```

```
[[ 0]
 [14]
 [ 1]
 [13]
 [19]
 [19]
 [13]]
```

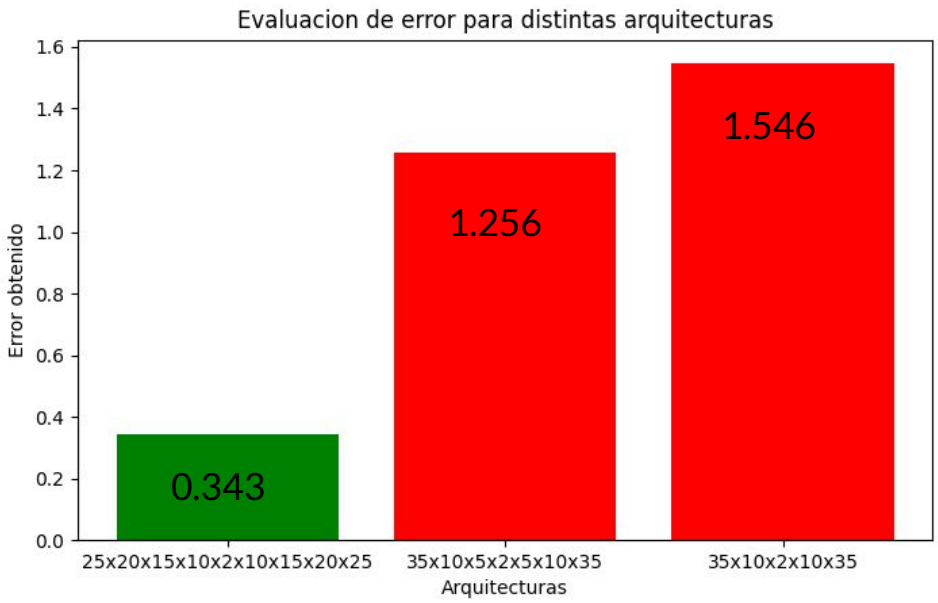
```
[[0 0 0 0 0]
 [0 1 1 1 0]
 [0 0 0 0 1]
 [0 1 1 0 1]
 [1 0 0 1 1]
 [1 0 0 1 1]
 [0 1 1 0 1]]
```



Conjunto de letras en “fonts.h”



Evaluación de la mejor arquitectura



Método: Powell

Learning rate = 0.0005

Iteraciones = 200

Todo el conjunto utilizado.

Elección de método de optimización

Se hicieron pruebas con la técnica de Adam y Powell.

El método de Adam fue descartado, ya que este nos convergía en menos de 20 iteraciones con un error de entre 12-15 sin importar la arquitectura usada.

Por lo tanto, teniendo en cuenta el buen performance de Powell se eligió este para la arquitectura final.

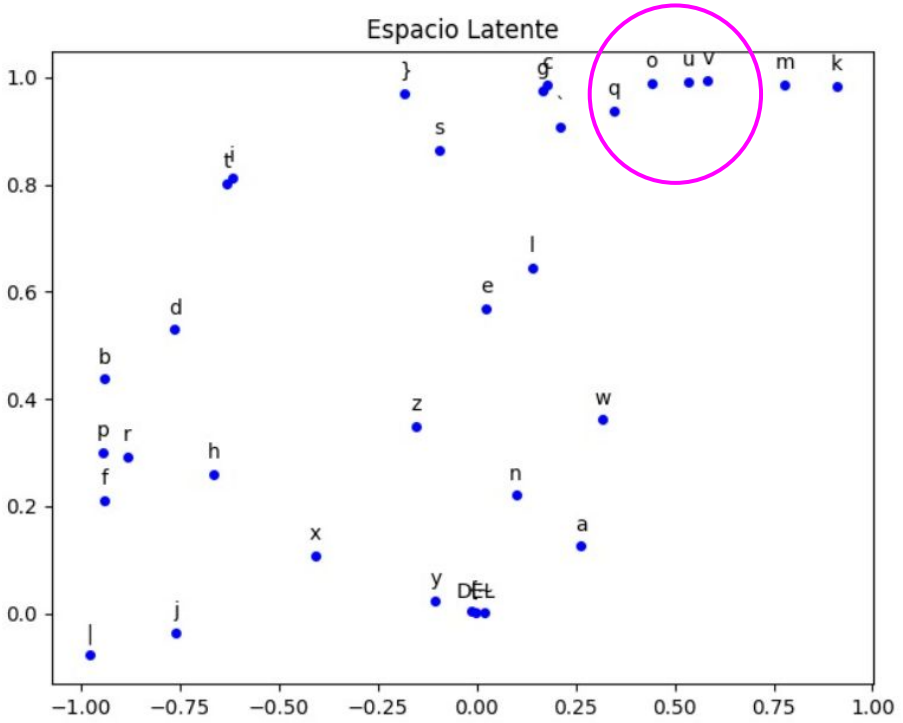
Resultados obtenidos

Método: Powell

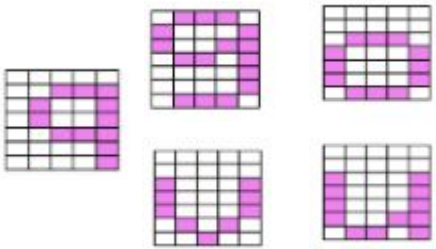
Capas = [25 20 15 10]

Iteraciones = 200

Todo el conjunto utilizado.



Ejemplo:



Resultados obtenidos

Método: Powell

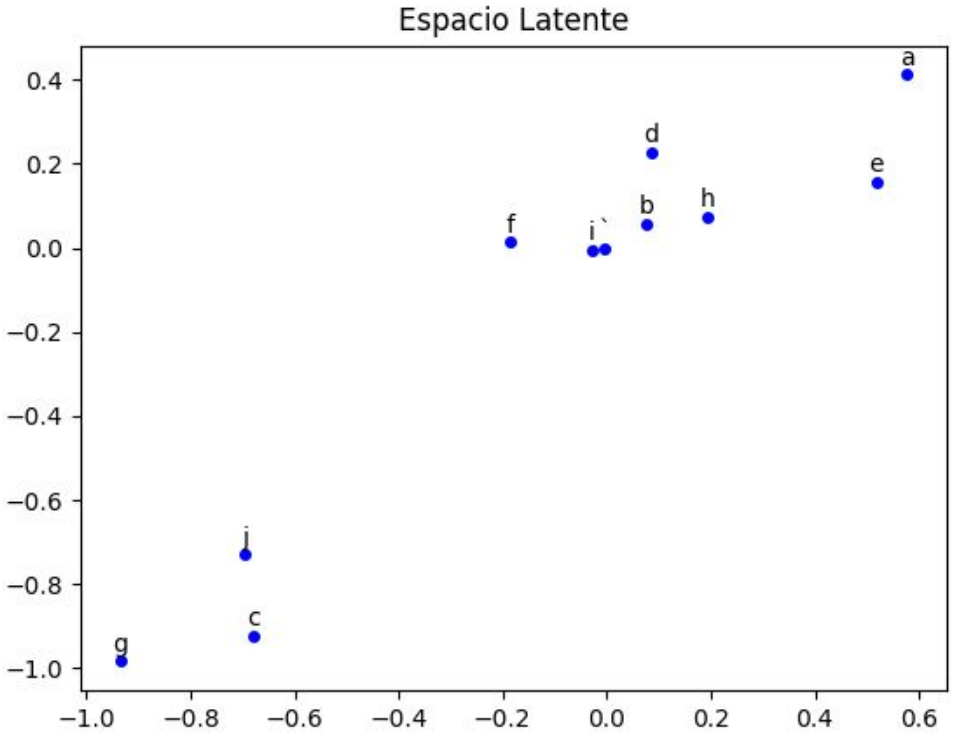
Capas = [25 20 15 10]

Iteraciones = 200

Conjunto utilizado:

[`, a, b, c, d, e, f, g, h, i, j]

Hubo un error de 0.2598



Resultados obtenidos

Creamos nuevas letras a partir de la mezcla aleatoria de 4 letras existentes y vemos si el Autoencoder es capaz de generarla

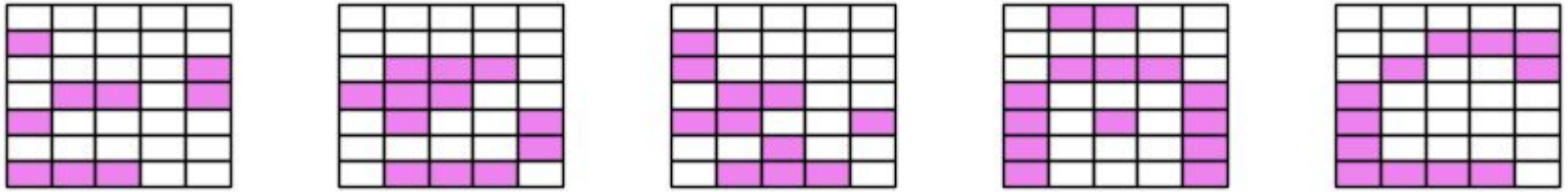
Método: Powell

Capas = [25 20 15 10]

Iteraciones = 200

Todo el conjunto utilizado.

Letras creadas:

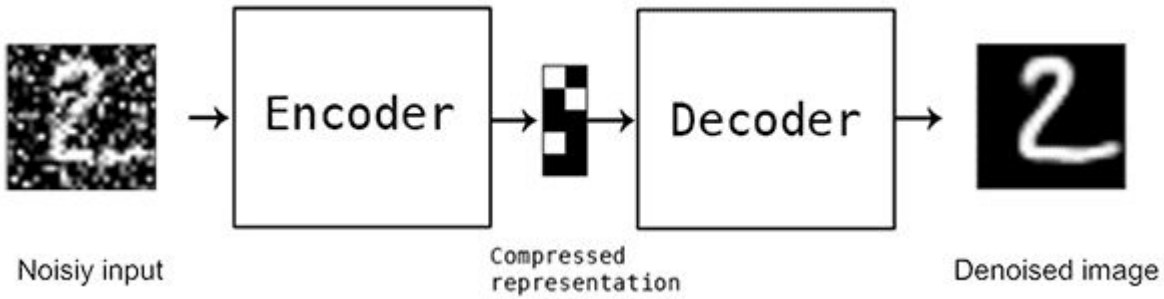


Se obtuvo un error de media de 1.213, demostrando que aunque no perfectamente, el Autoencoder es más que capaz de generar patrones con los que no fue entrenado

Denoising Autoencoder

Denoising Autoencoder

- Es un autoencoder que es entrenado con las entradas con ruido y se busca eliminarlo en la salida.
- Su objetivo es reconstruir la información con ruido.



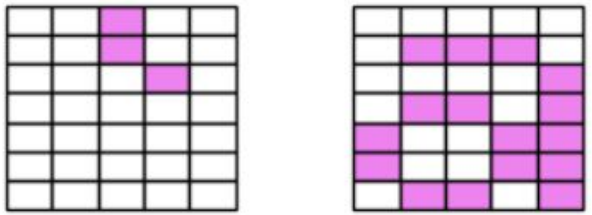
Ejercicio 1.b

Sobre el mismo dataset, implementar una variante que implemente un ”Denoising Autoencoder”.

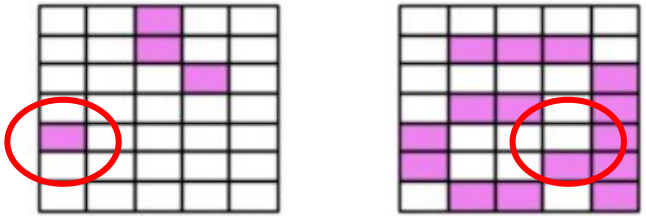
1. Plantear una arquitectura de red conveniente para esta tarea. Explicar la elección.
2. Distorsionen las entradas en diferentes niveles y estudien la capacidad del Autoencoder de eliminar el ruido.

Mutación de patrones

RUIDO 0



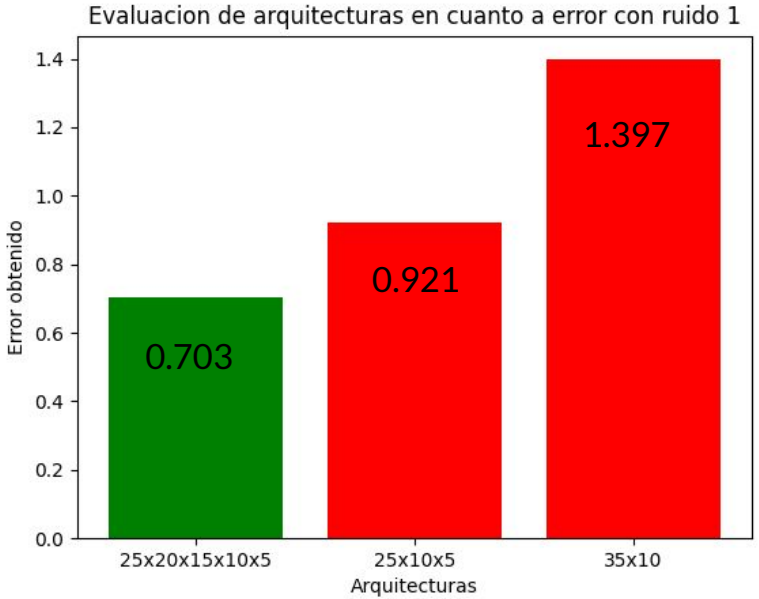
RUIDO 1



RUIDO 2

Evaluación de arquitecturas

RUIDO 1

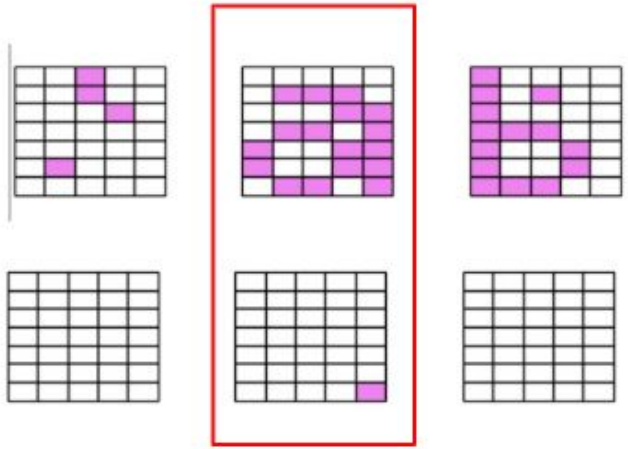


Método: Powell **Learning rate = 0.0005**

Iteraciones = 50 **Conjunto = [`, a, b, c, d, e, f, g, h, i]**

Evaluación del muestreo de ruido para más precisión

RUIDO 1

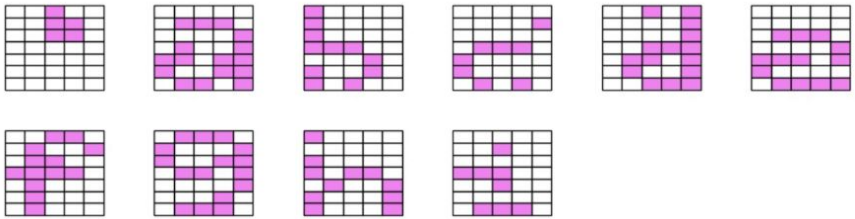


```
[-2.63651438e-02  4.80681411e-01  6.33385839e-01  5.50243863e-01
 -3.04738837e-02  5.43941815e-01  1.99204125e-01  6.35704170e-02
  3.81955684e-02  8.74304068e-01  9.11712953e-01 -1.44435532e-01
  2.49988707e-02  7.89529995e-01  9.79565696e-01  9.99645716e-01
  9.99648808e-01  9.95055115e-01  1.78938455e-01  8.60042700e-01
  1.76893977e-01  2.05047586e-01 -3.90715932e-02  1.54284339e-01
  9.56556554e-01  8.92995561e-02  2.46887390e-01  8.14397720e-02
  1.47238371e-01  9.56513655e-01 -2.48525524e-01  7.77048257e-01
  9.64128599e-01  8.40273277e-01  1.00000000e+00]
```

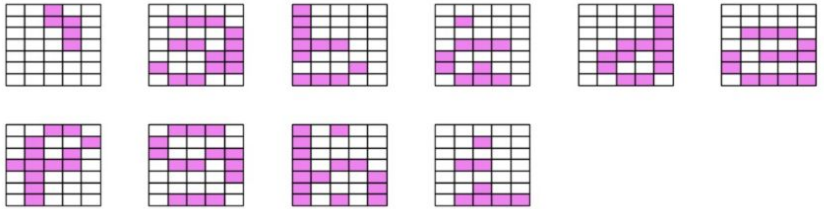
Evaluación del muestreo de ruido para más precisión

RUIDO 1

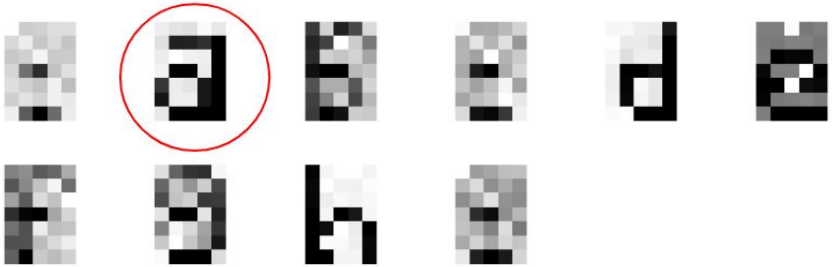
Entrenamiento:



Conjunto de entrada:



Conjunto de salida:

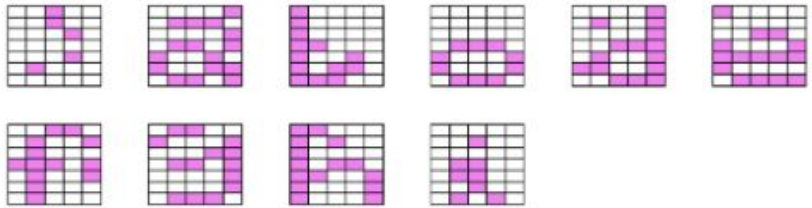



```
[ 1.09001430e-02  5.26097370e-02  6.49646617e-02 -4.16534875e-02
  3.21244188e-02  7.01004415e-02  8.60834622e-01  8.53892167e-01
  7.98908032e-01  9.99996496e-01  2.15895904e-02 -6.09856934e-02
 -5.85900052e-04  6.71481841e-03  9.99999990e-01 -8.59759152e-03
  1.00000000e+00  1.00000000e+00  1.96334369e-02  9.99999997e-01
  7.72054298e-01  1.90063661e-02  4.80011415e-02  8.93570966e-01
  9.98584858e-01  9.10645014e-01  1.00365318e-02  1.16970028e-02
  8.96693907e-01  1.00000000e+00  3.24790258e-02  9.99712346e-01
  1.00000000e+00  1.00000000e+00  9.99866595e-01]
```

Error = 1.327

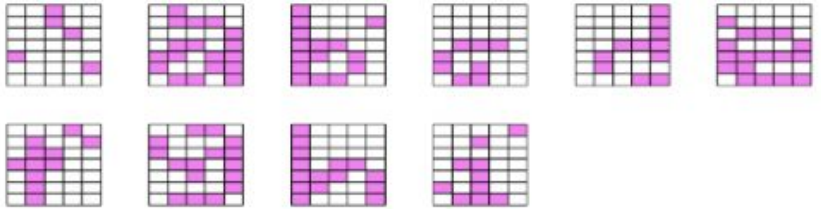
Resultados obtenidos con ruido 2

Entrenamiento:

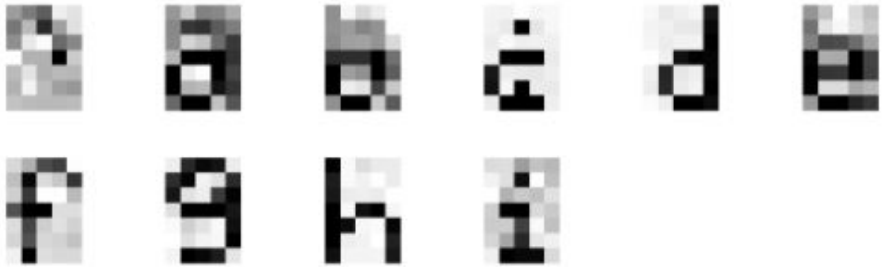


Error = 0.959
Iteraciones = 50
Capas = [25 20 15 10 5]

Conjunto de entrada:

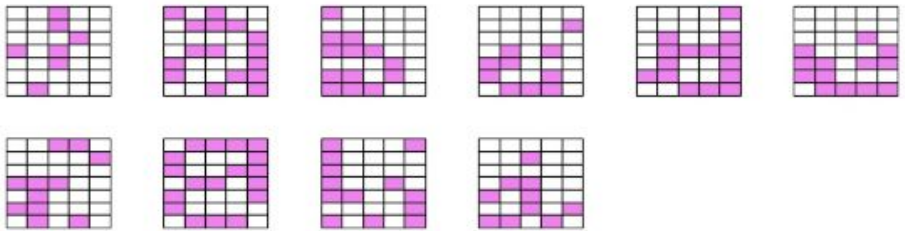


Conjunto de salida:

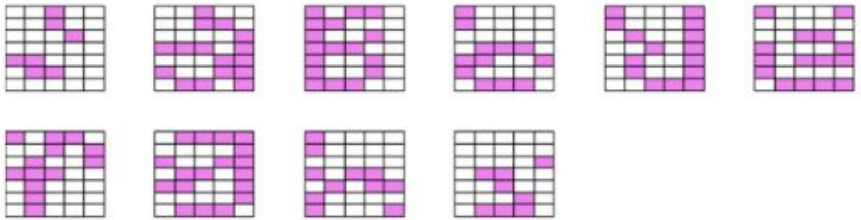


Resultados obtenidos con ruido 4

Entrenamiento:

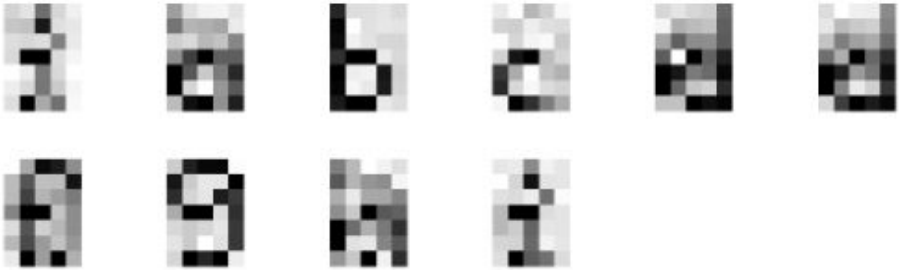


Conjunto de entrada:



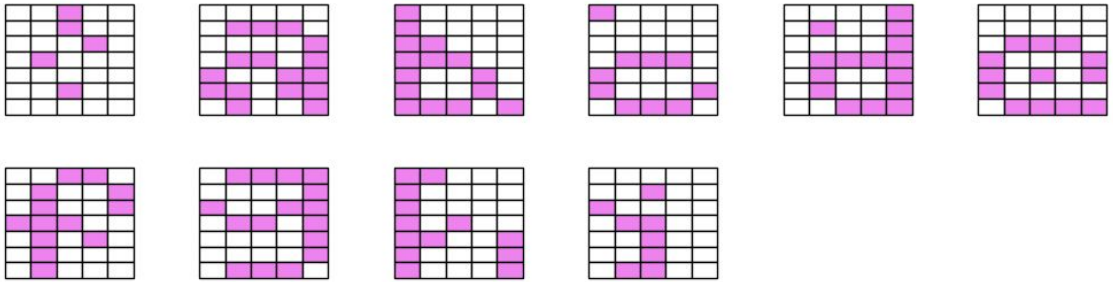
Error = 1.337
Iteraciones = 50
Capas = [25 20 15 10 5]

Conjunto de salida:



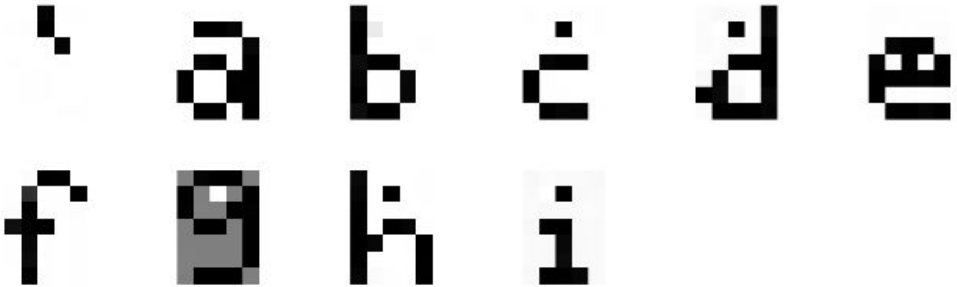
Resultado obtenido con ruido 2 con mismo conjunto de entrenamiento y entrada

Conjunto de entrenamiento y entrada:



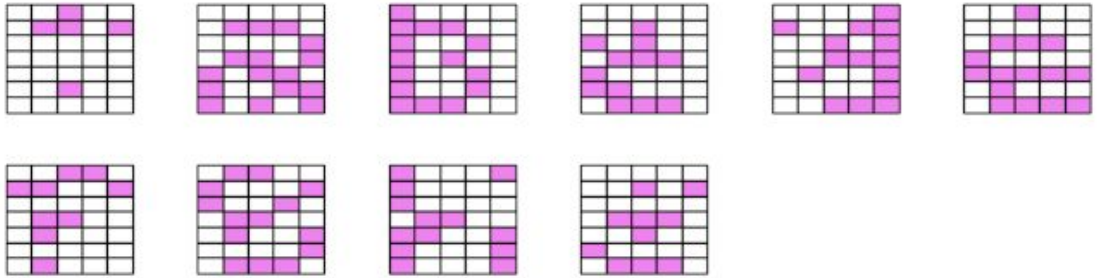
Error = 0.2546
Iteraciones = 50
Capas = [25 20 15 10 5]

Conjunto de salida:



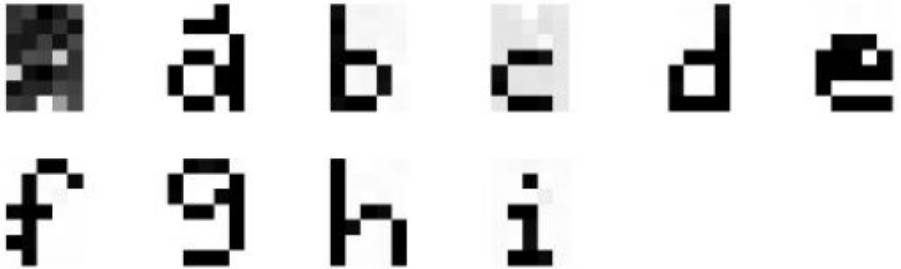
Resultado obtenido con ruido 4 con mismo conjunto de entrenamiento y entrada

Conjunto de entrenamiento y entrada:



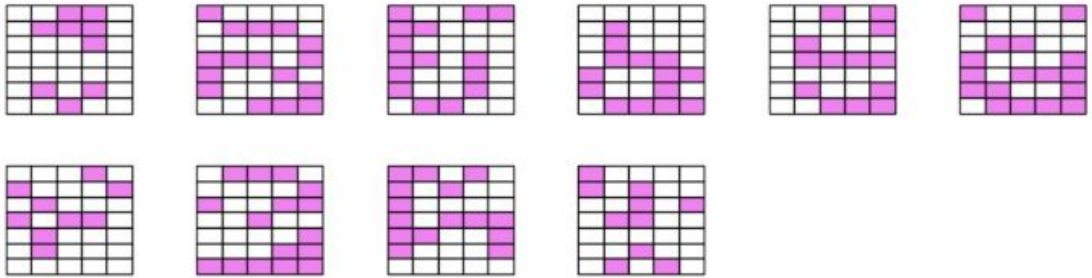
Error = 0.3544
Iteraciones = 50
Capas = [25 20 15 10 5]

Conjunto de salida:

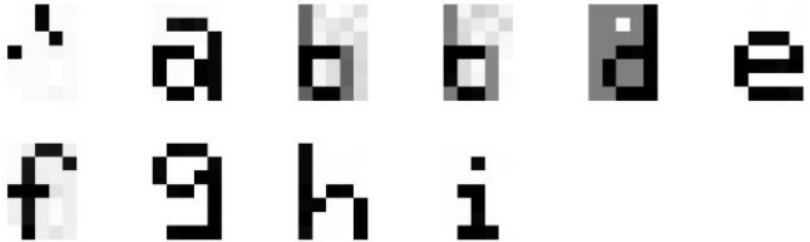


Resultado obtenido con ruido 6 con mismo conjunto de entrenamiento y entrada

Conjunto de entrenamiento y entrada:



Conjunto de salida:



Error = 0.447
Iteraciones = 50
Capas = [25 20 15 10 5]

Conclusiones

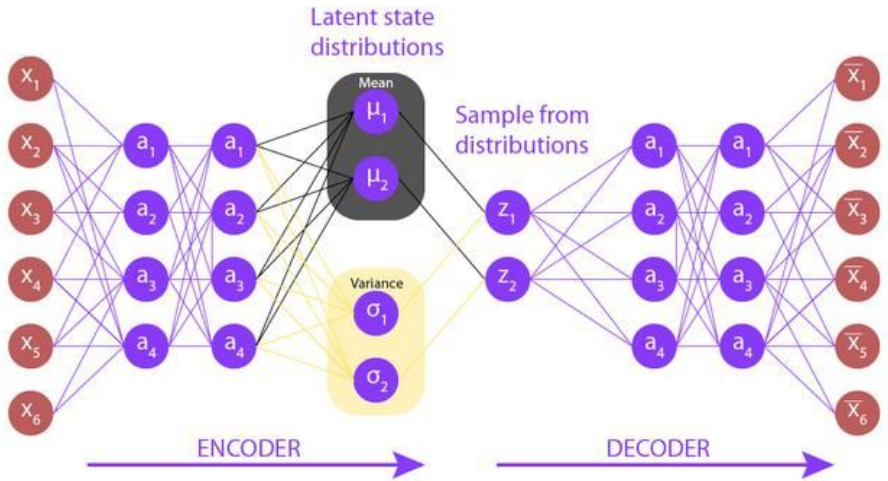
Con todas las pruebas realizadas pudimos sacar las siguiente conclusiones:

- Aunque tome más tiempo entrenar, el autoencoder es un herramienta muy poderosa para aprender muy bien a reconocer patrones y reducir la dimensionalidad de esto.
- Es eficiente a la hora de encontrar anomalías luego de su entrenamiento.
- Si aplicamos mucho ruido, los resultados son malos.
- Cuando se presenta un patrón con ruido distinto a la capa de entrada al autoencoder ya entrenado con ruido distinto, no es capaz de eliminarlo de la mejor forma.

Variational Autoencoder

Variational Autoencoder

- La red aprende una distribución de probabilidades que modela información.
- Obteniendo puntos de esta distribución, se puede generar nuevos ejemplos, y gracias a esto podemos decir que un VAE es un *modelo generativo*

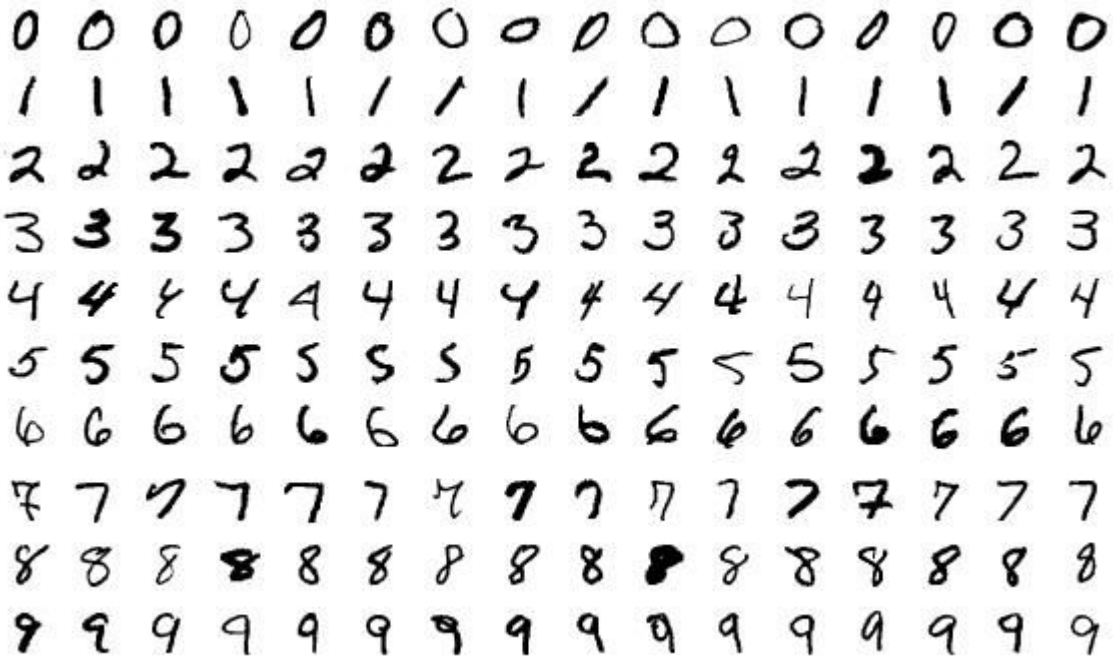


Ejercicio 2

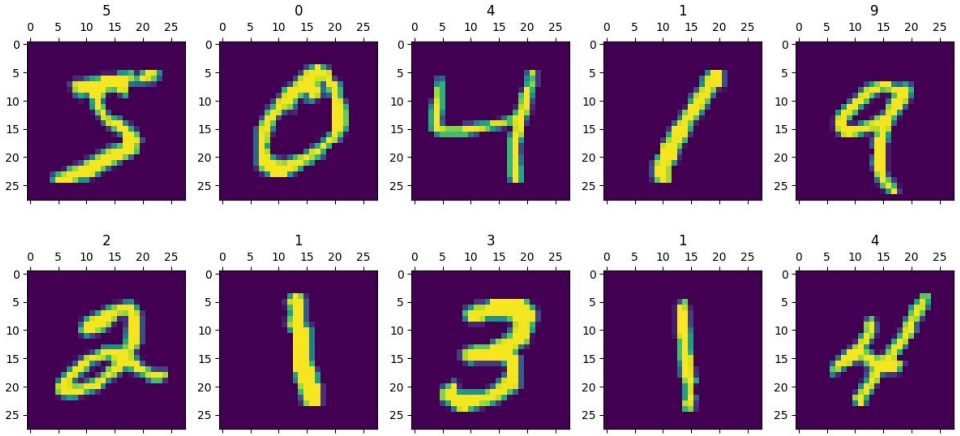
Dada la capacidad generativa variacional del autoencoder,

1. Elegir (o construir) un conjunto de datos nuevos (por ejemplo emojis) y utilizar el Autoencoder anterior para generar una nueva muestra que ustedes puedan juzgar que pertenece al conjunto de datos que le presentaron al autoencoder.
2. Modificar el autoencoder planteando un esquema variacional para poder solucionar el problema de la representación en el espacio latente.

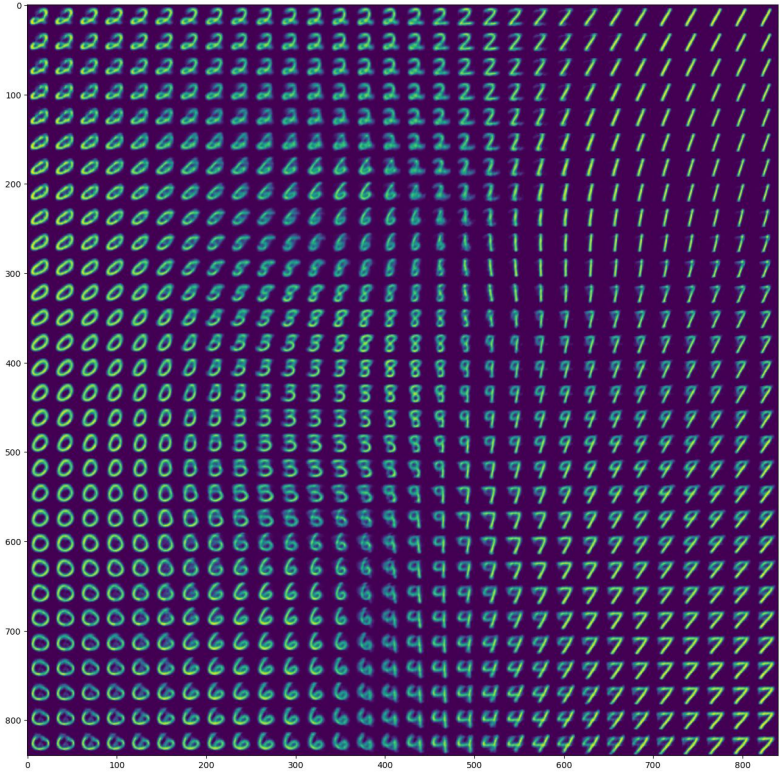
Minst dataset



Resultados obtenidos

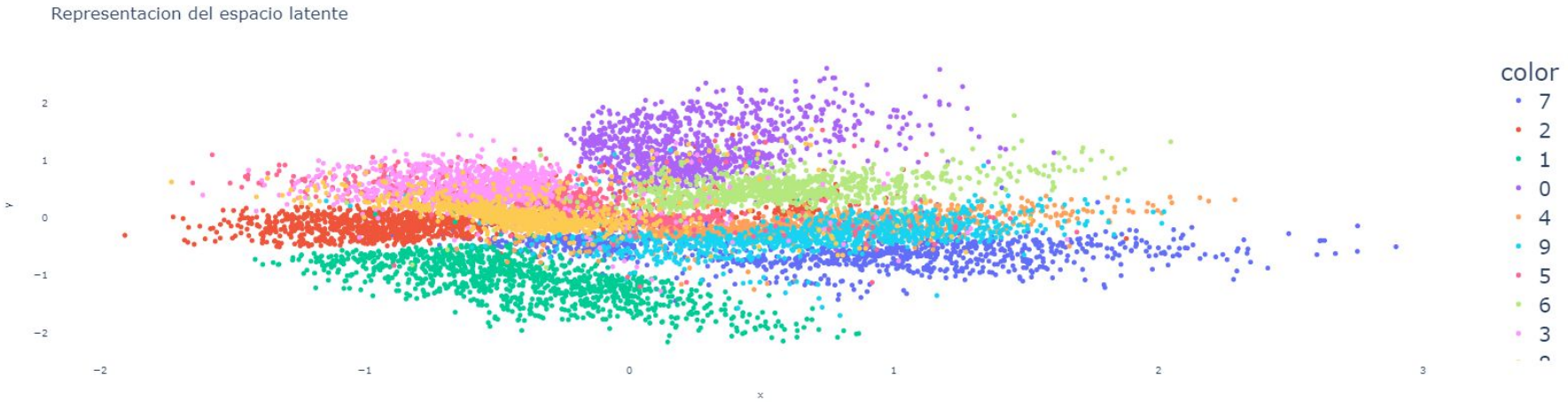


MNIST



Resultados obtenidos

Representación del espacio latente



Capas:[784,64,16,8]
Épocas:25

Conclusiones

- Un Autoencoder Variacional tiene la capacidad de generar nuevas muestras dentro del espacio latente. Esto se logra muestreando puntos aleatorios en el espacio latente y decodificándolos en el espacio de entrada para generar nuevas muestras.
- Al desplazarse en la capa latente, se pueden generar nuevas muestras que son una combinación de características presentes en otras muestras. Esto permite generar variaciones y mezclas de patrones existentes.
- Los nuevos modelos generados por un VAE se basan en los patrones y estructuras aprendidas durante el entrenamiento. A través del aprendizaje del espacio latente y la capacidad de muestreo, el VAE puede generar nuevas muestras que siguen las distribuciones y características de los datos de entrenamiento.
- Para el correcto funcionamiento de un VAE, es recomendable que el conjunto de entrada esté normalizado. Esto ayuda a asegurar que los datos estén en una escala y rango adecuados para el aprendizaje y la generación de muestras precisas.

Fin.