SIA-TP5

Deep Learning

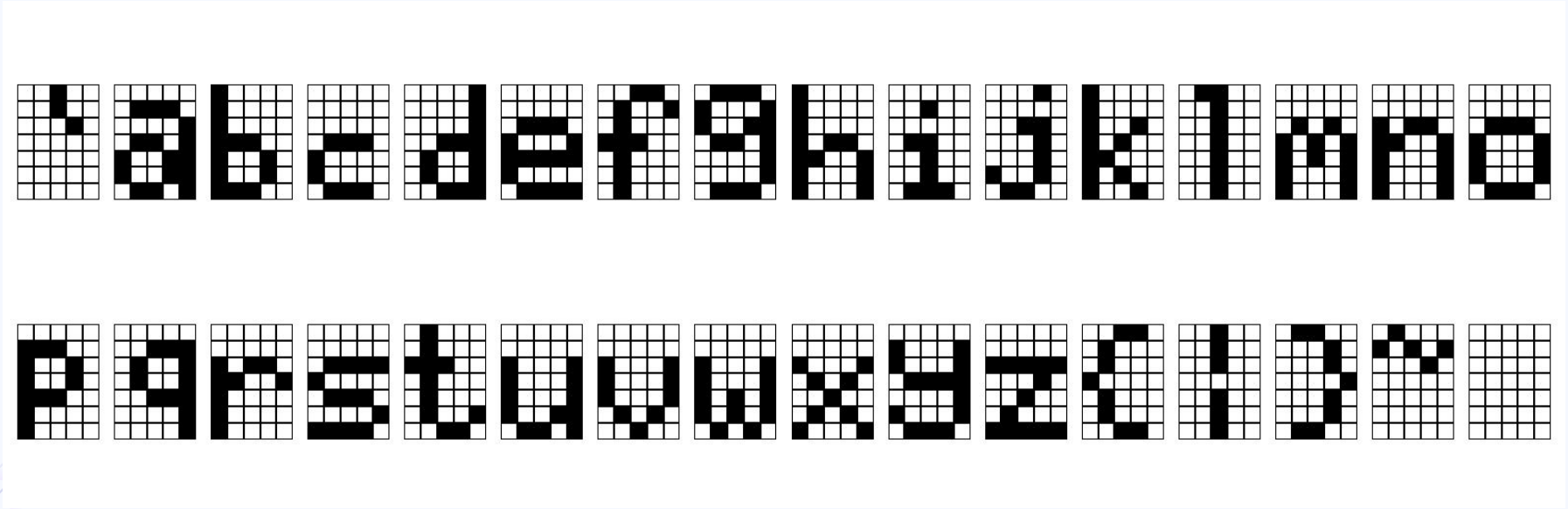
Grupo 6

- Francisco Sendot
- Lucia Digon
- Martín E. Zahnd
- Juan Ignacio Fernández Dinardo

01

Autoencoder

Lista de caracteres “font.h”



Búsqueda de arquitectura: variando

Capas

Probamos distintas cantidades de capas de distintos tamaños

Epochs

Variamos la cantidad de epochs

Learning rate

Cambiamos el learning rate

Optimizadores

Probamos con Adam y Gradient Descent

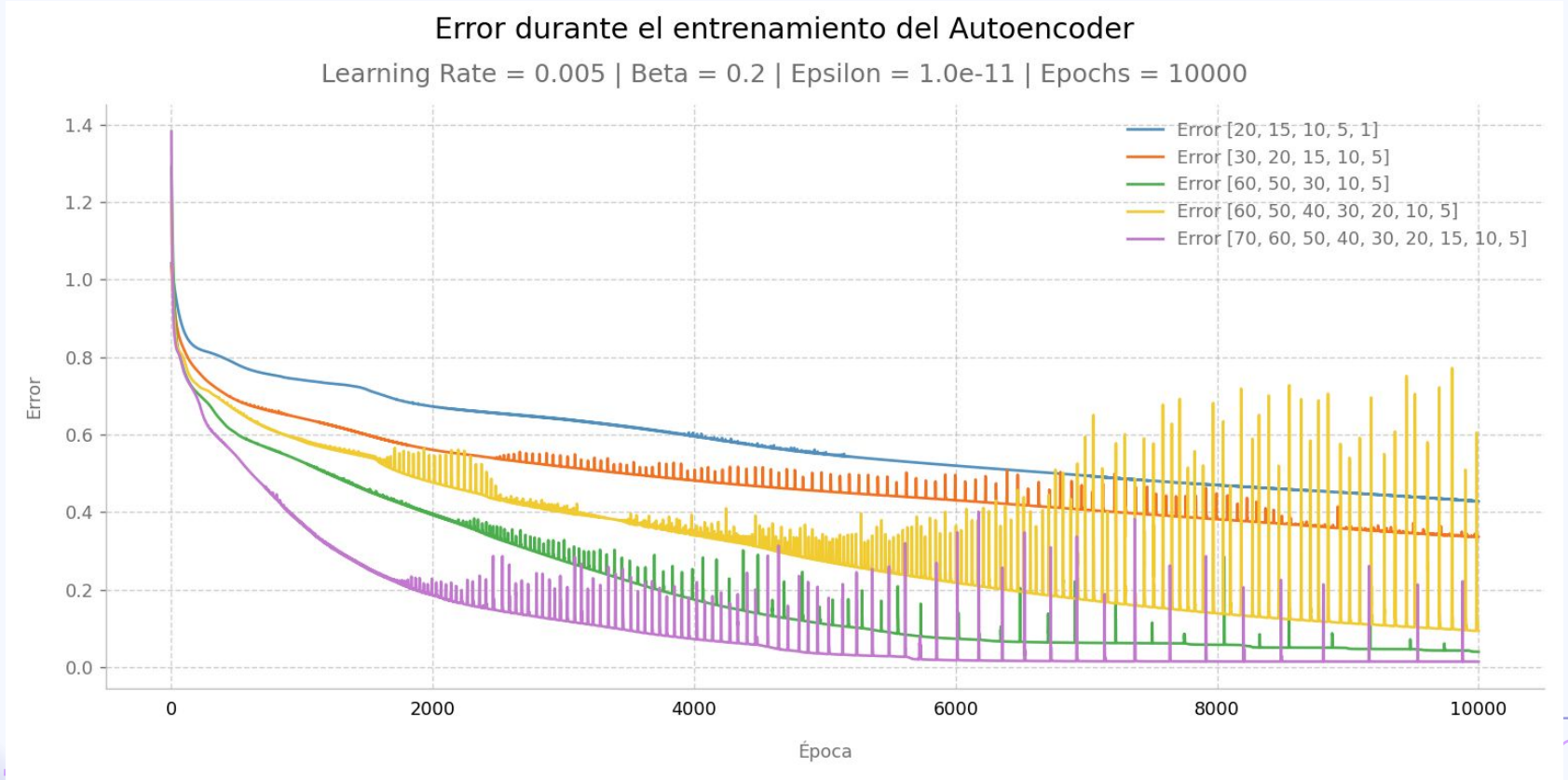
Func. activación

Tanh, sigmoid, logistic

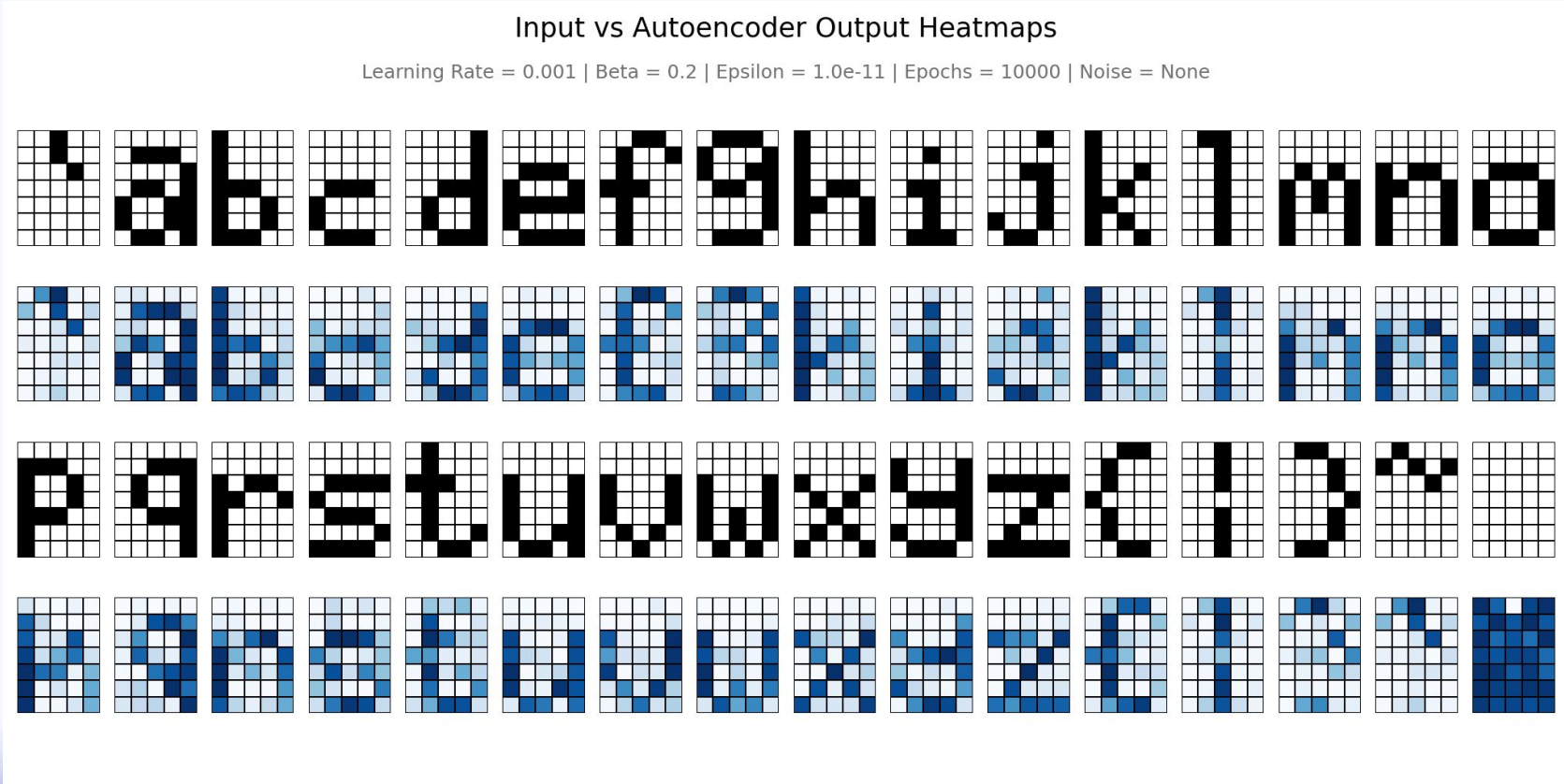
Entrenamiento

Batch y mini batch

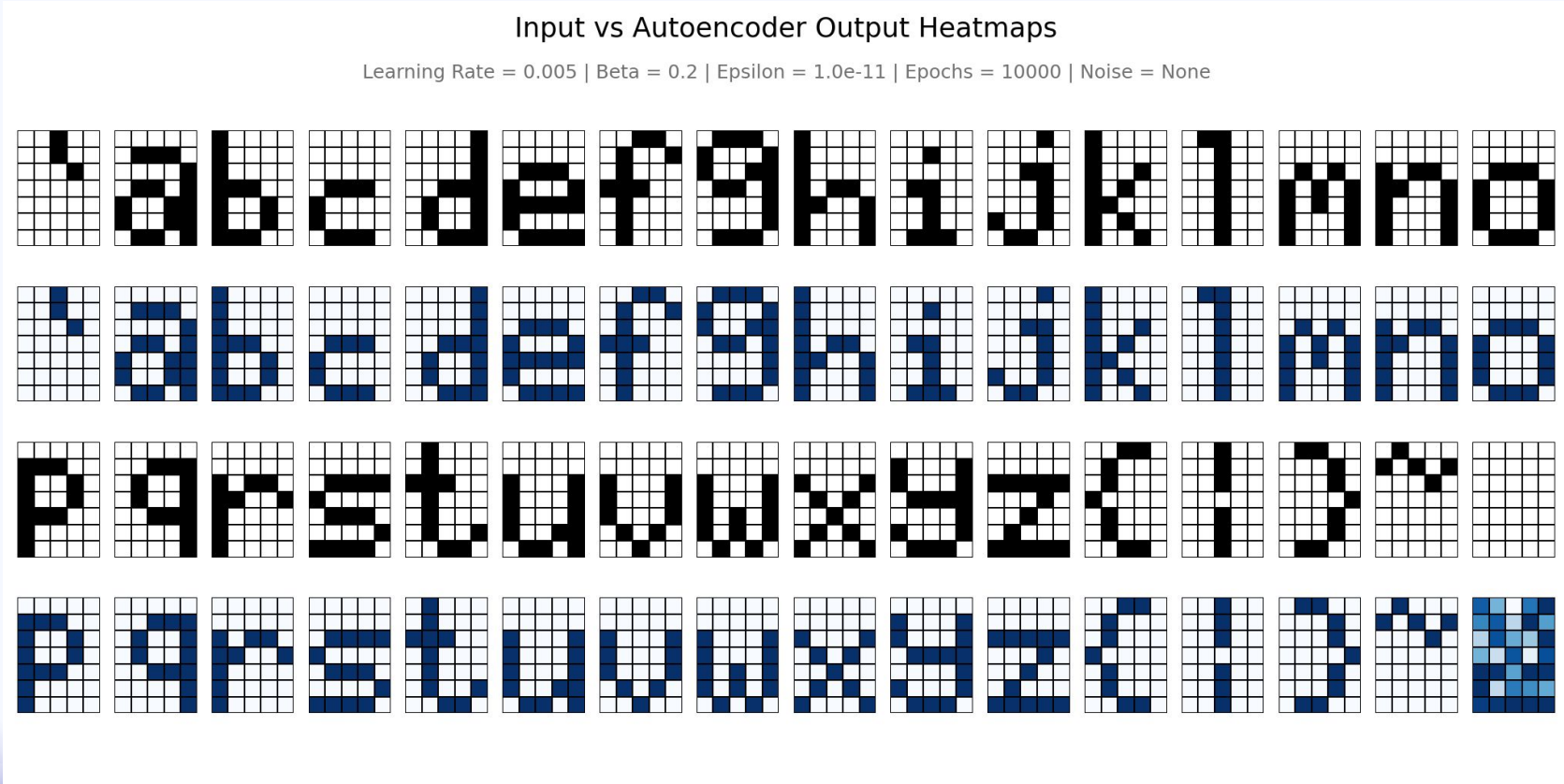
Variando arquitectura (capas)



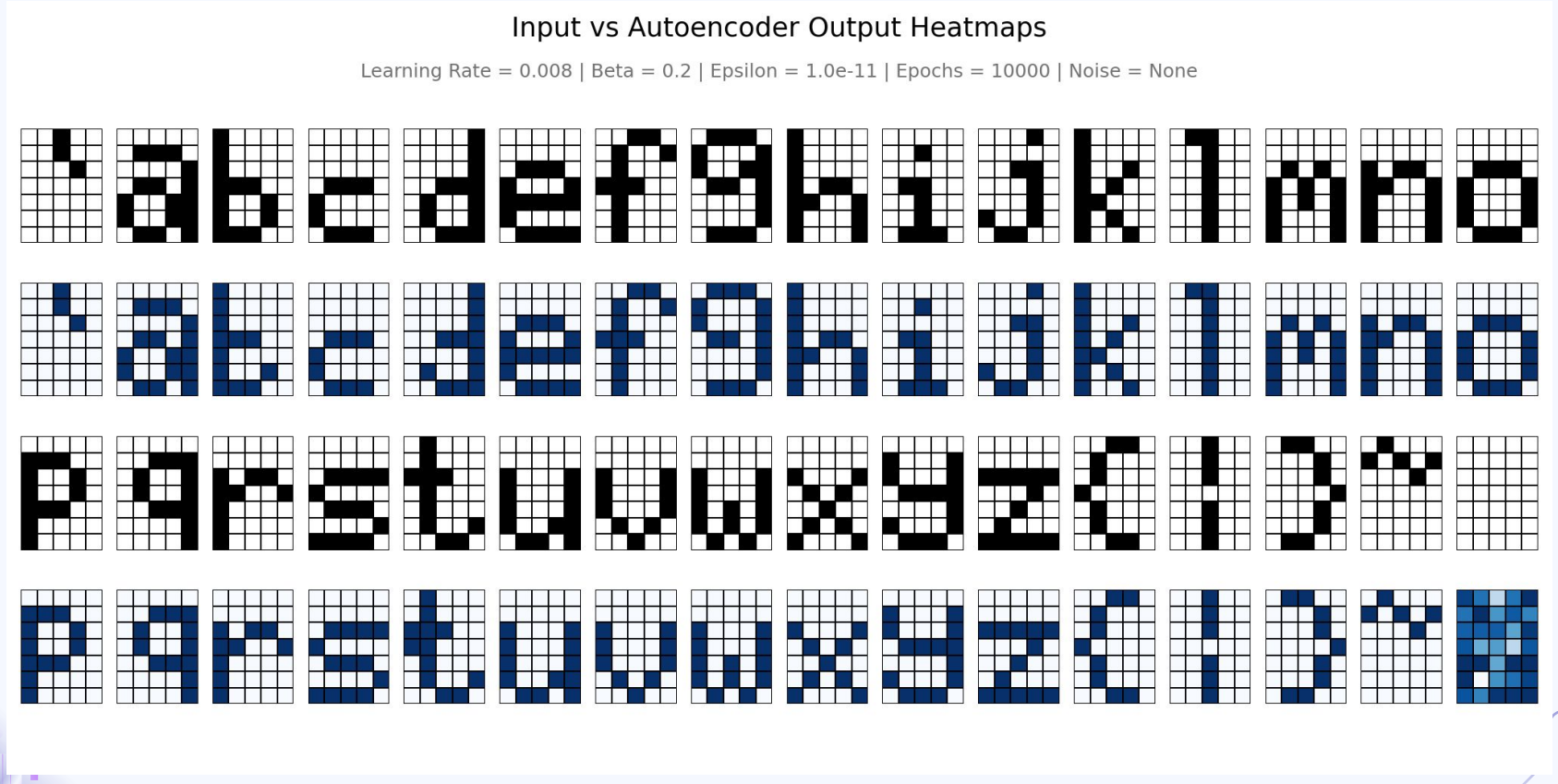
Variando learning rate: 0.001



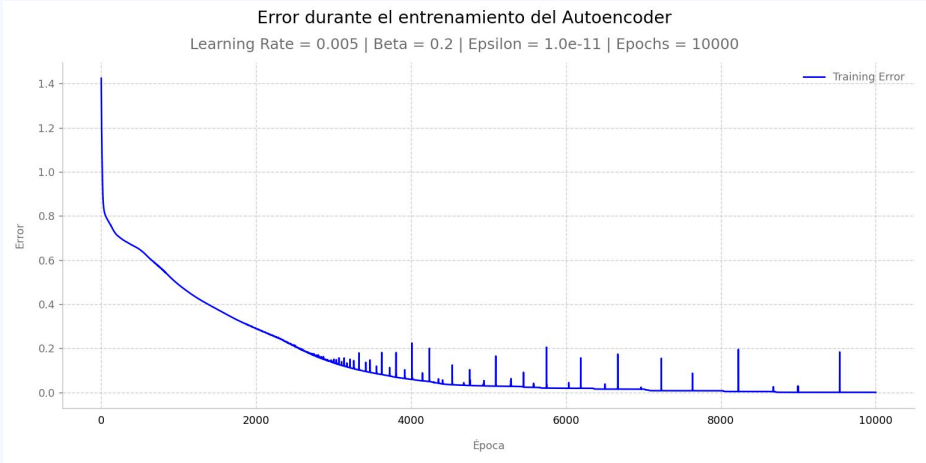
Variando learning rate: 0.005

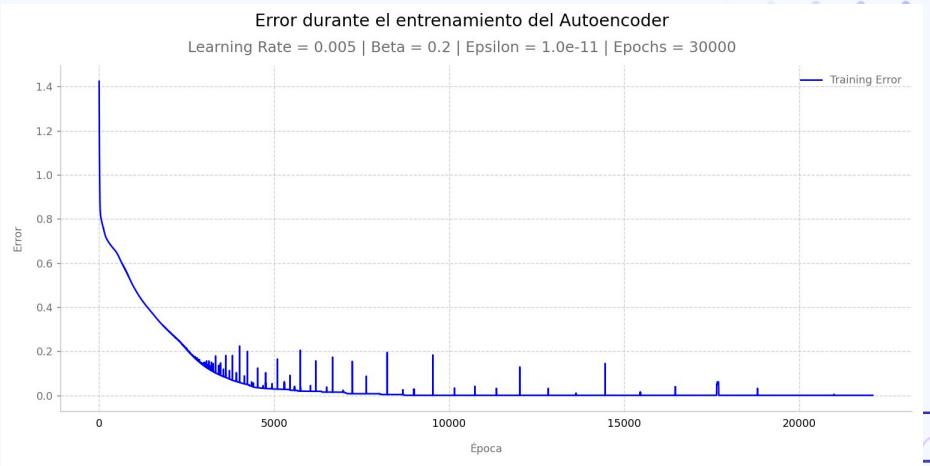


Variando learning rate: 0.008



Variando Epochs

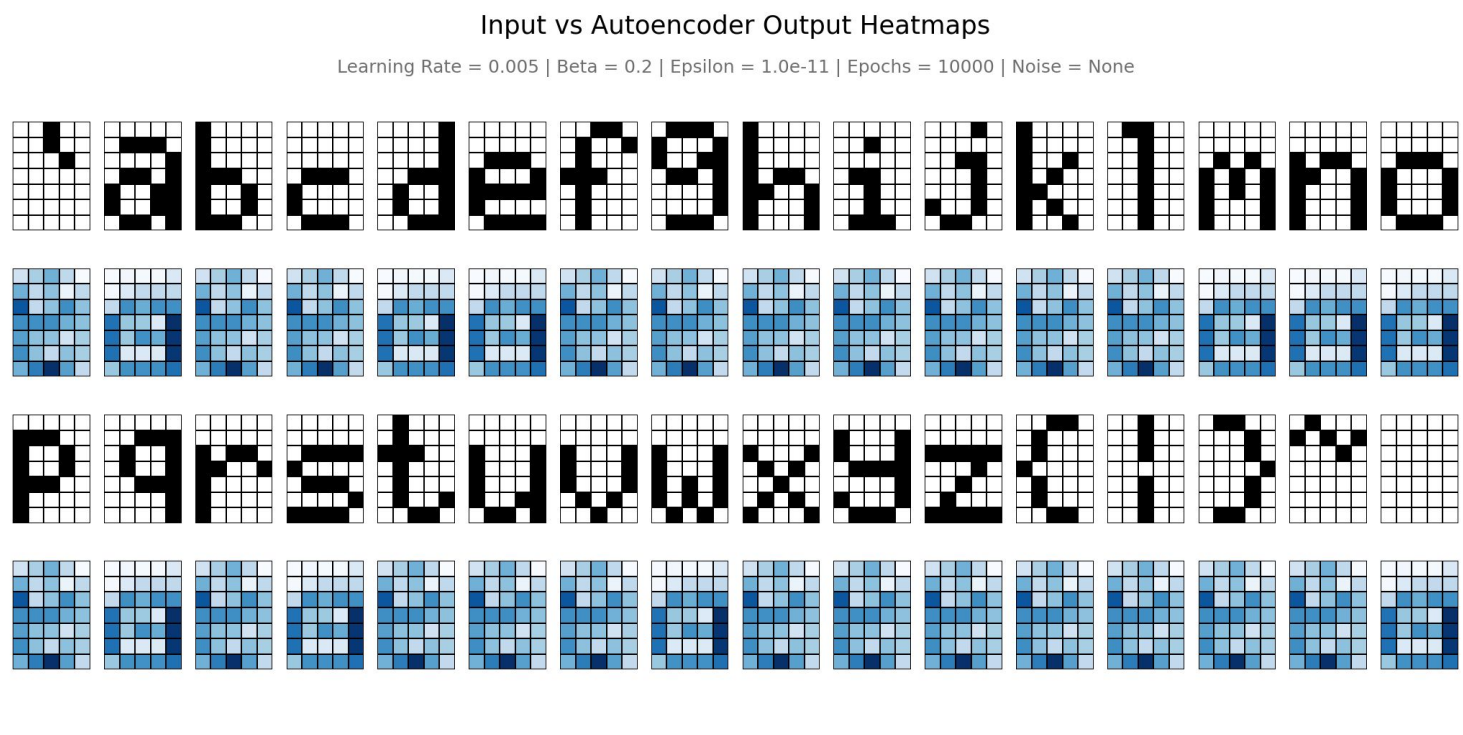




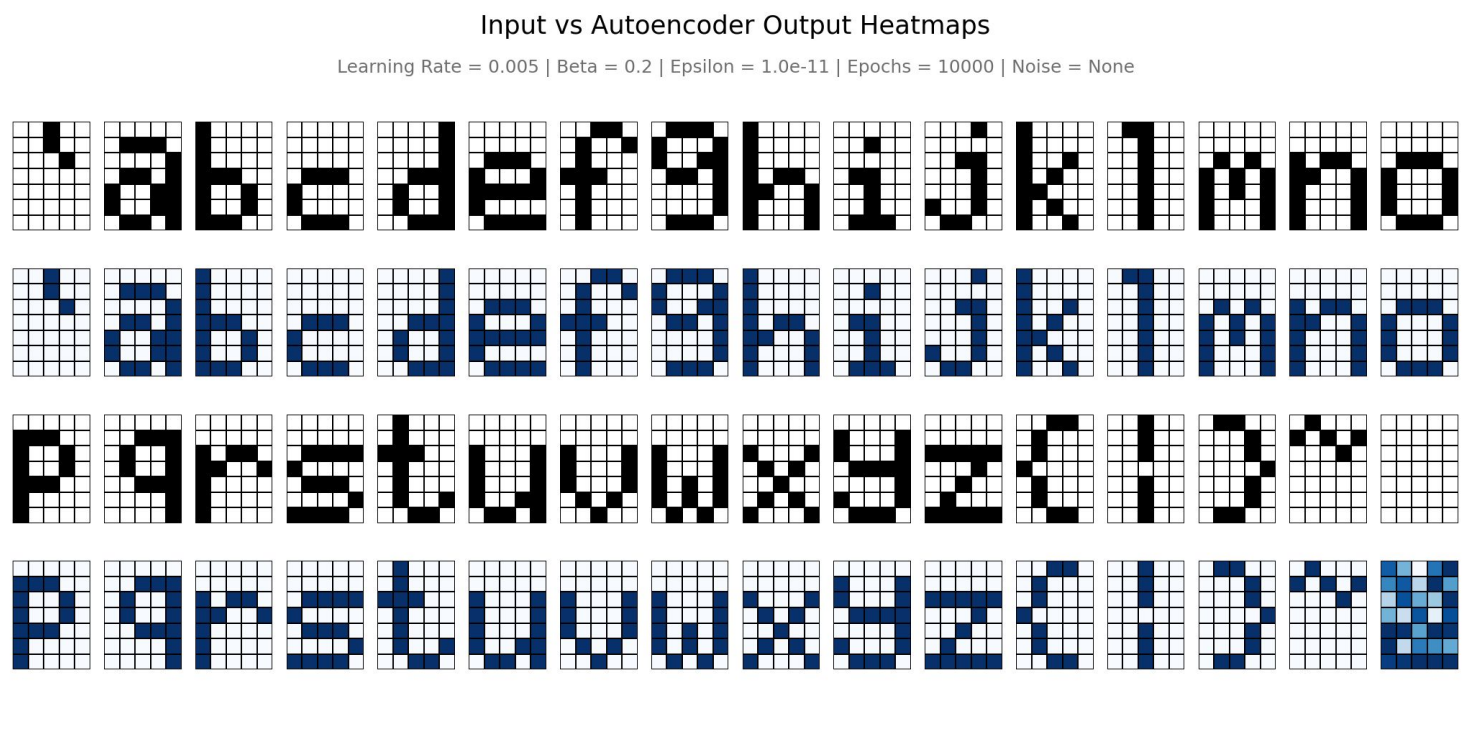
10.000

30.000

Variando Optimizadores: Gradient Descent



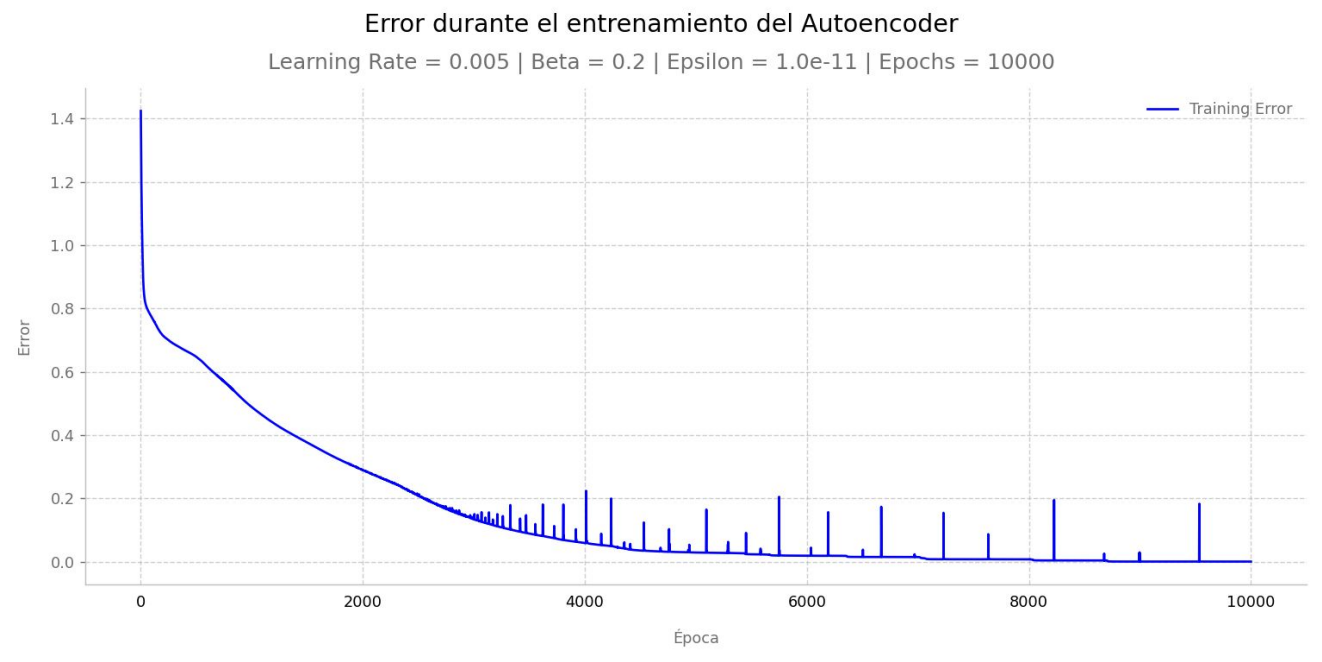
Variando Optimizadores: Adam



Arquitectura elegida

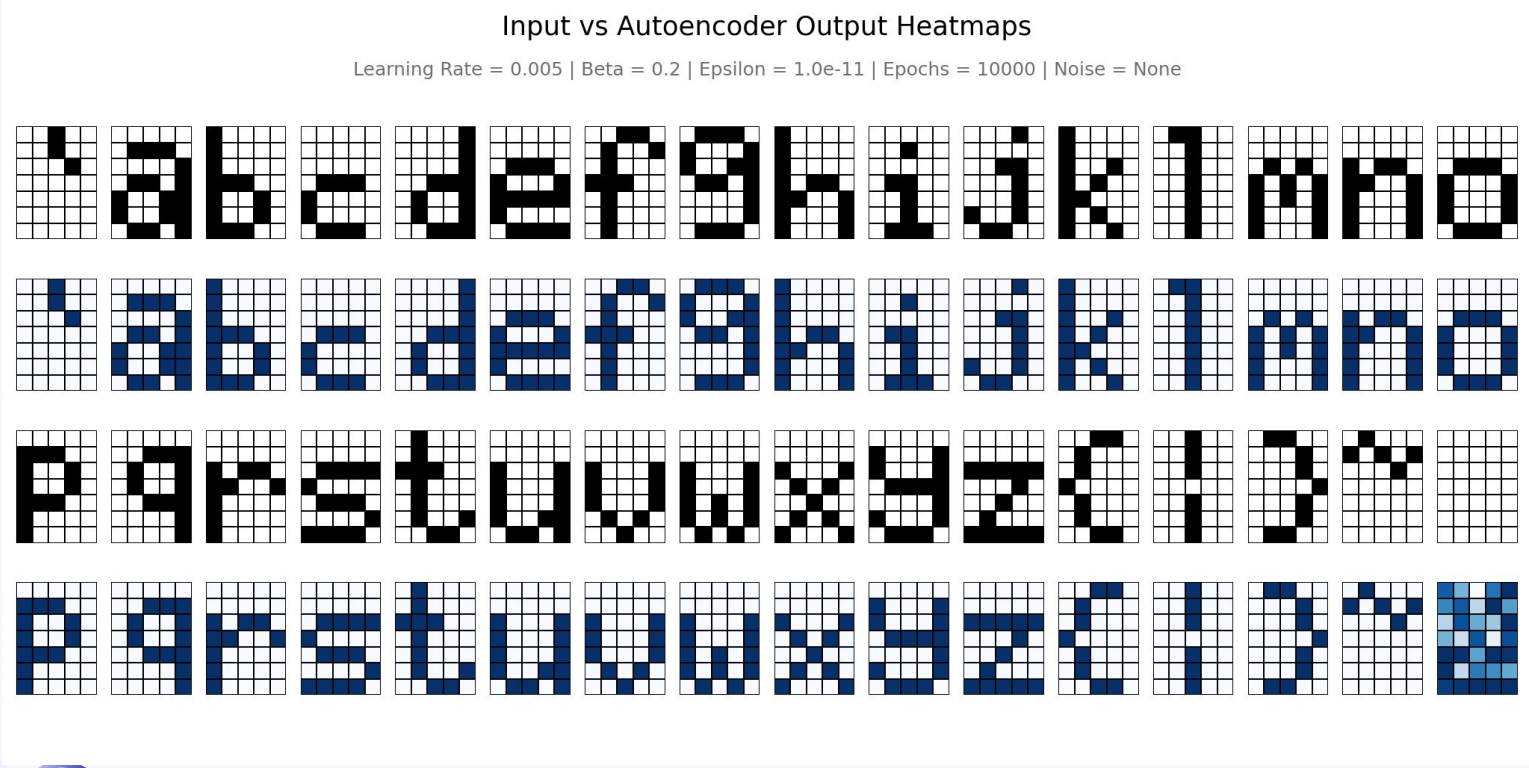
35-60-50-30-10-5-2-5-10-30-50-60-35

- **Epochs:** 10.000
- Tanh
- Adam
- **Learning rate:** 0,005

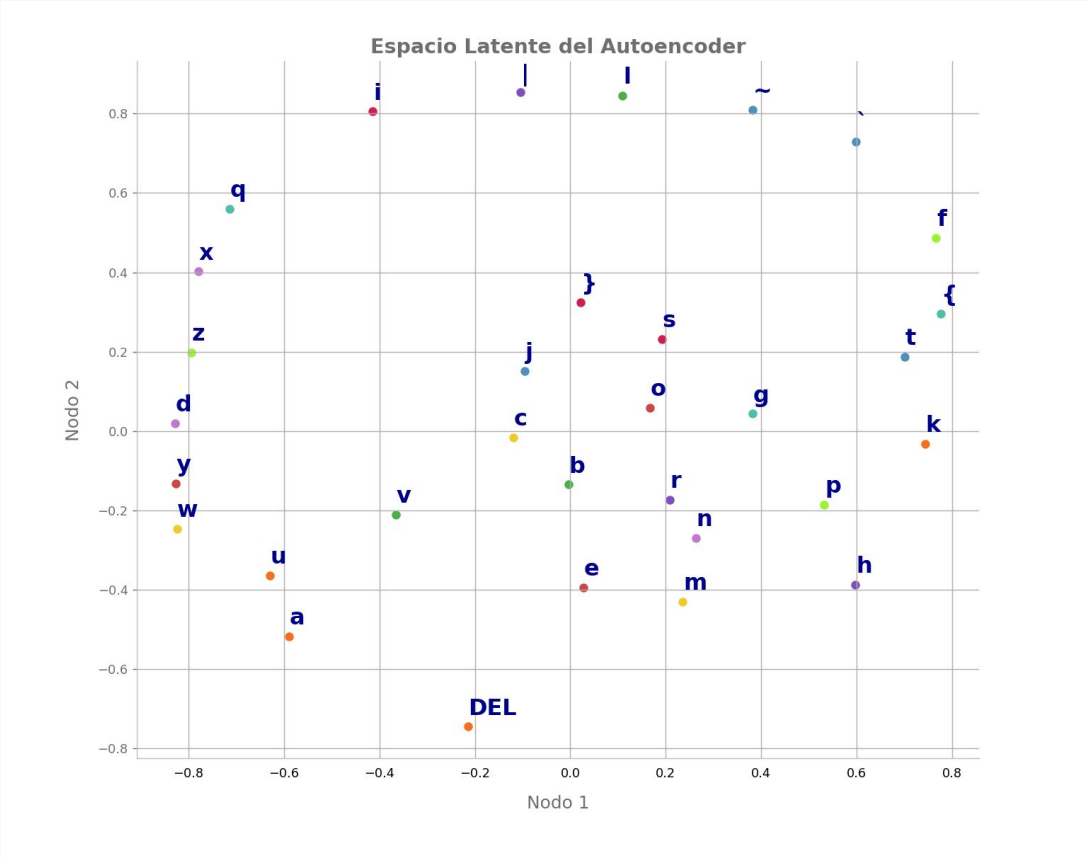


Arquitectura elegida

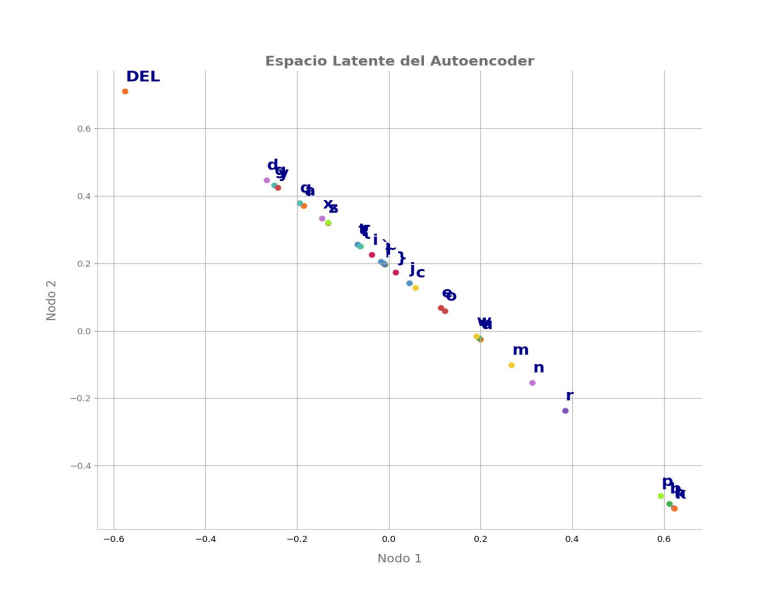
35-60-50-30-10-5-2-5-10-30-50-60-35



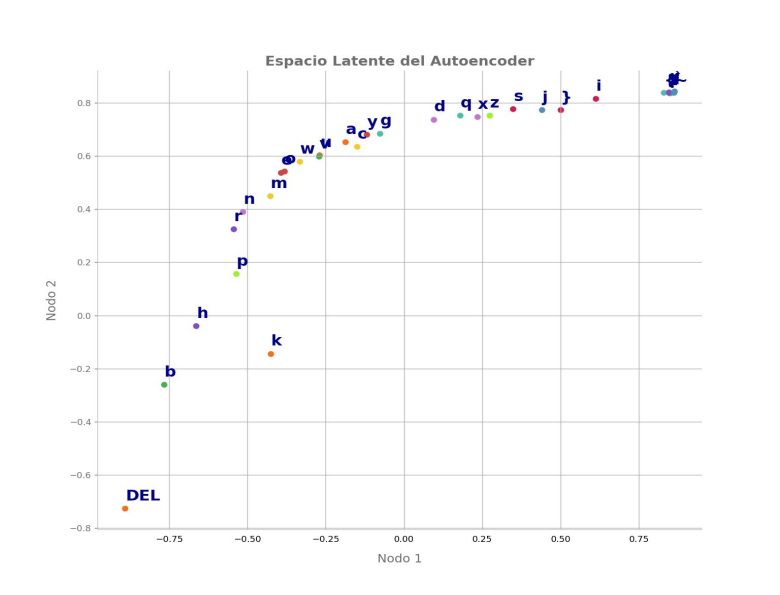
Datos de entrada en el espacio latente



Espacios latentes para arq que no aprendieron

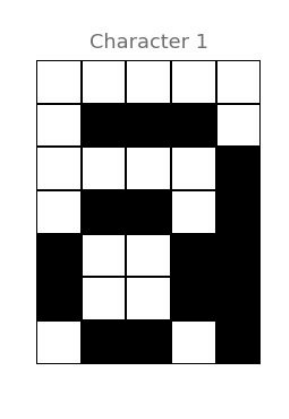


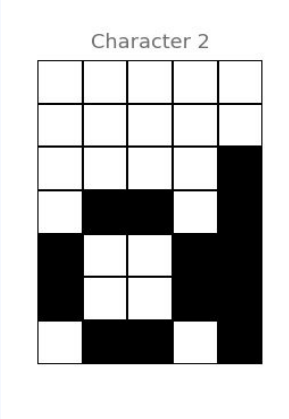
35-20-15-10-5-1-2

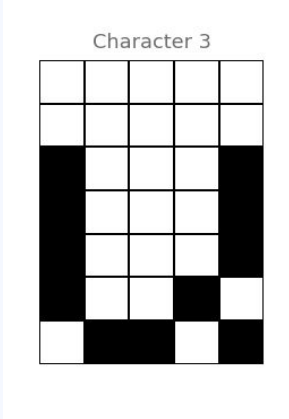


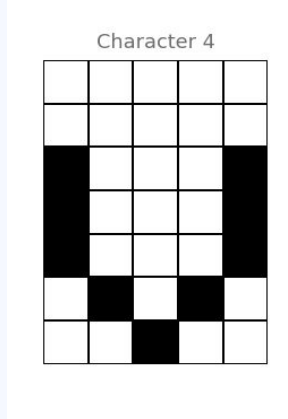
35-30-20-15-10-5-2

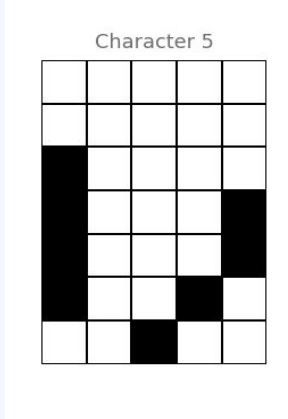
Generación nueva letra

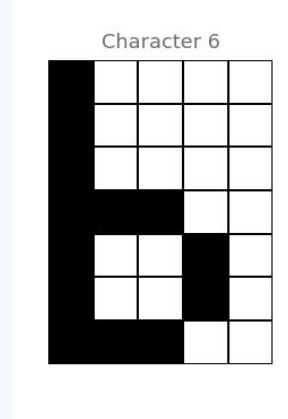




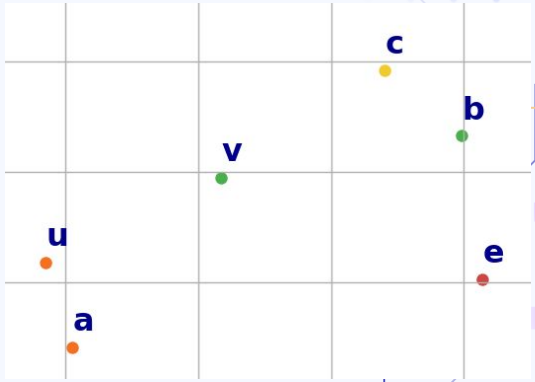








- Se obtiene el la codificación dentro del espacio latente de dos caracteres, en este caso a y b
- Se toma un “paso”, y se recorre el espacio latente desde uno a otro.



Conclusiones

- Para diferentes arquitecturas, varía el espacio latente, más si no llega a aprender bien.
- En el espacio latente, se puede ver las letras que tienen codificaciones similares
- Llega un punto que seguir bajando el learning rate hace que el autoencoder no llegue a aprender bien

02 Denoising Autoencoder

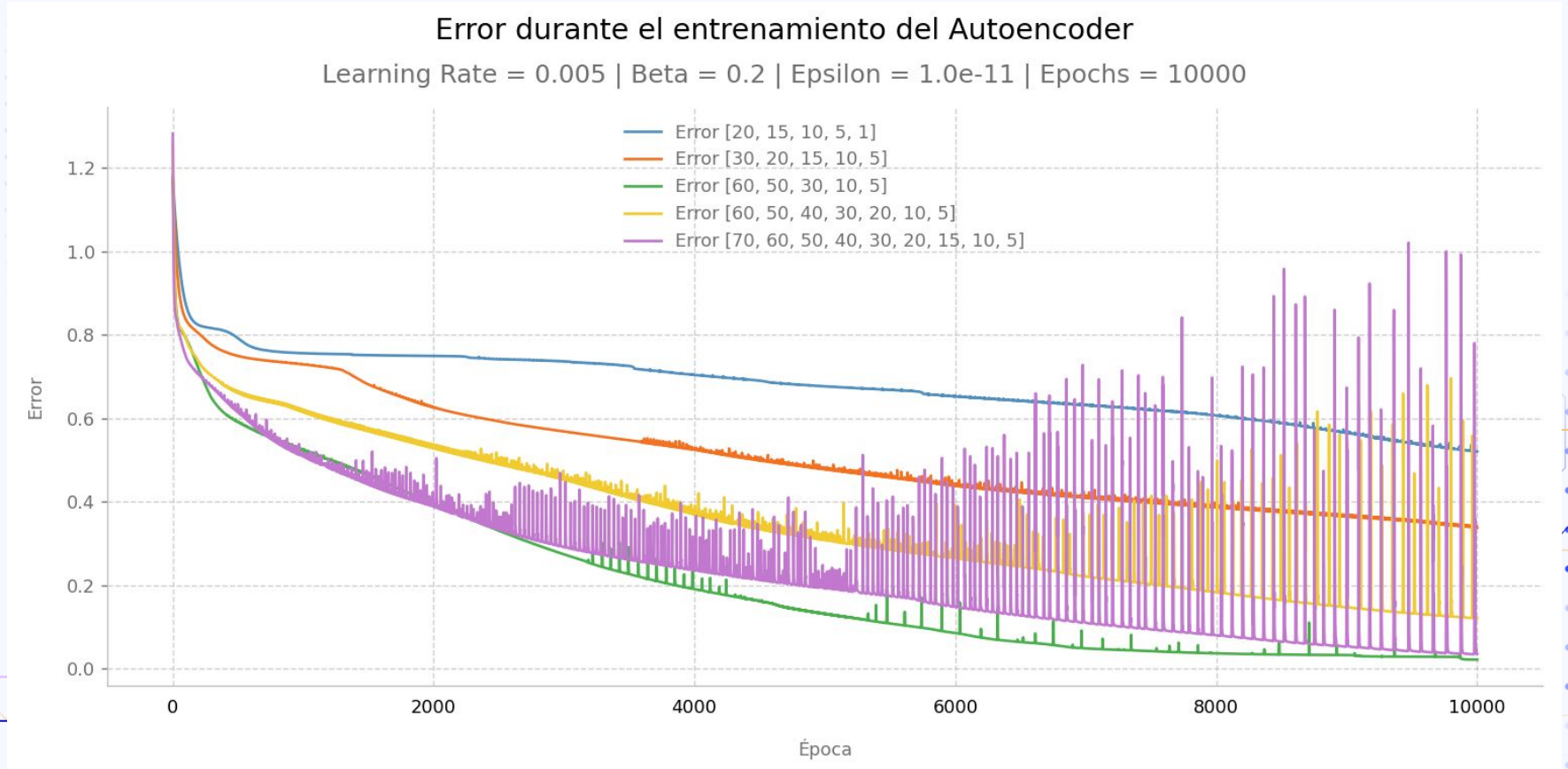
Ruido

- Utilizamos ruido Gaussiano
- Tomamos cada pixel prendido y alteramos los de su entorno siguiendo

$$\mathcal{N}(\mu, \sigma^2) * \text{intensity} / 2$$

Con $\mu = 1$ y $\sigma^2 = 1$

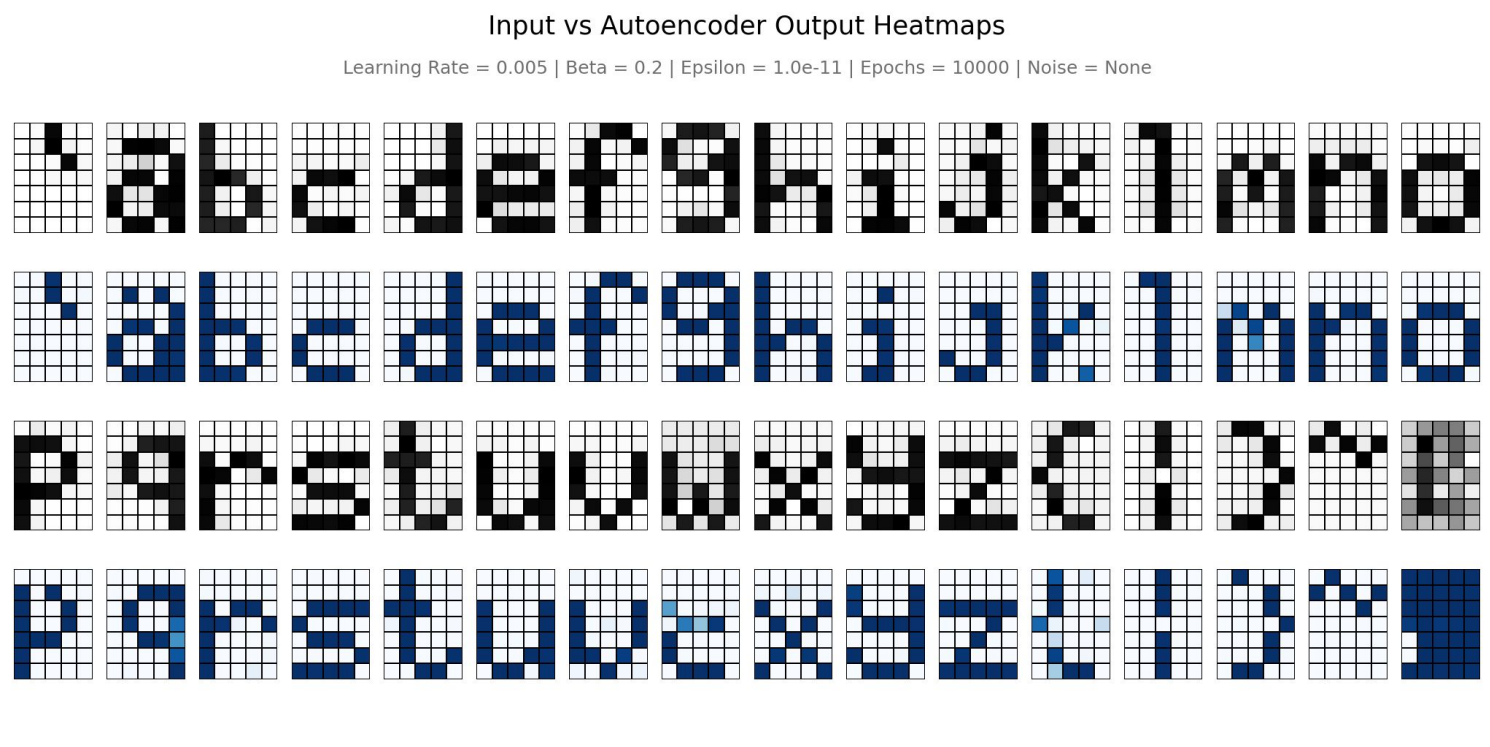
Ruido - Arquitectura



Dataset - Ruido 0.1

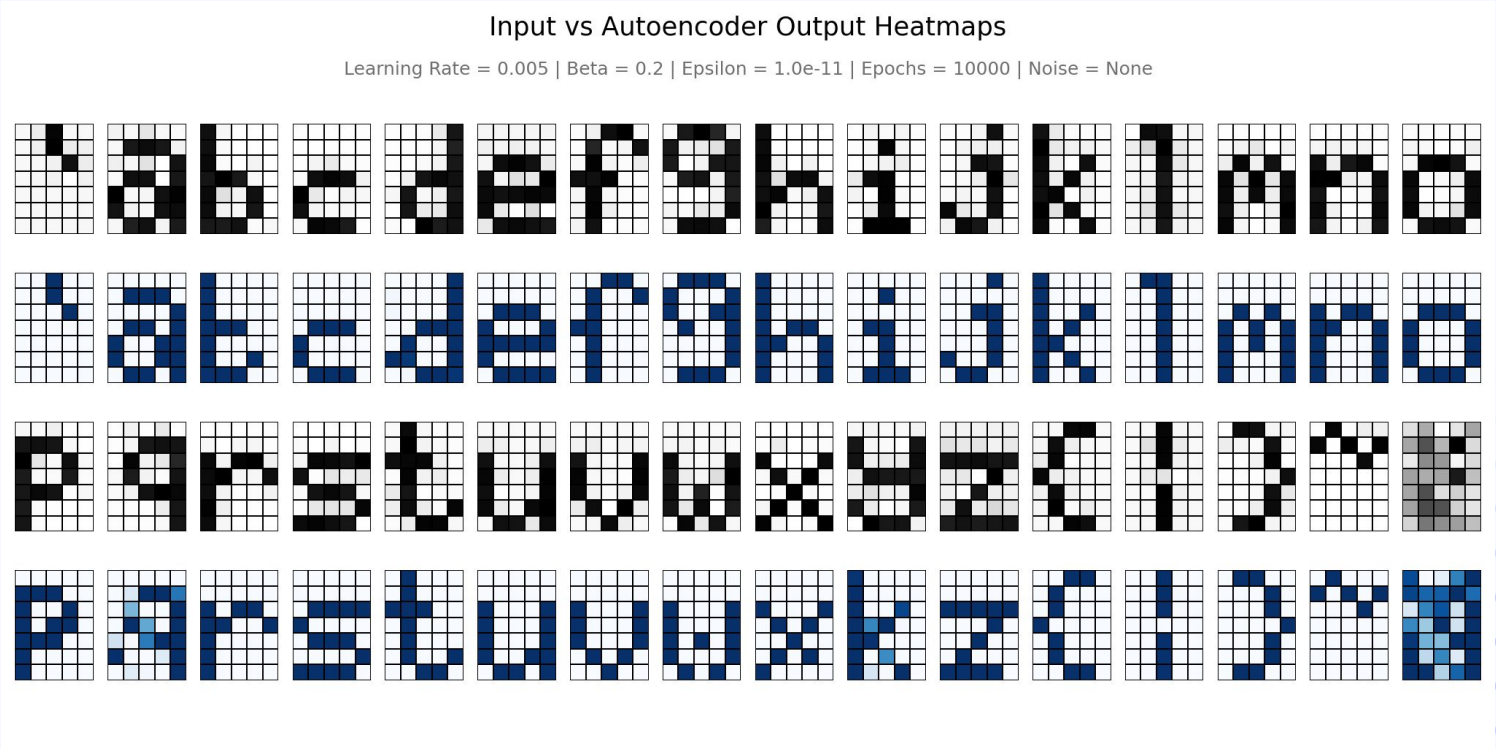
Entrada con ruido del entrenamiento

Aprendizaje



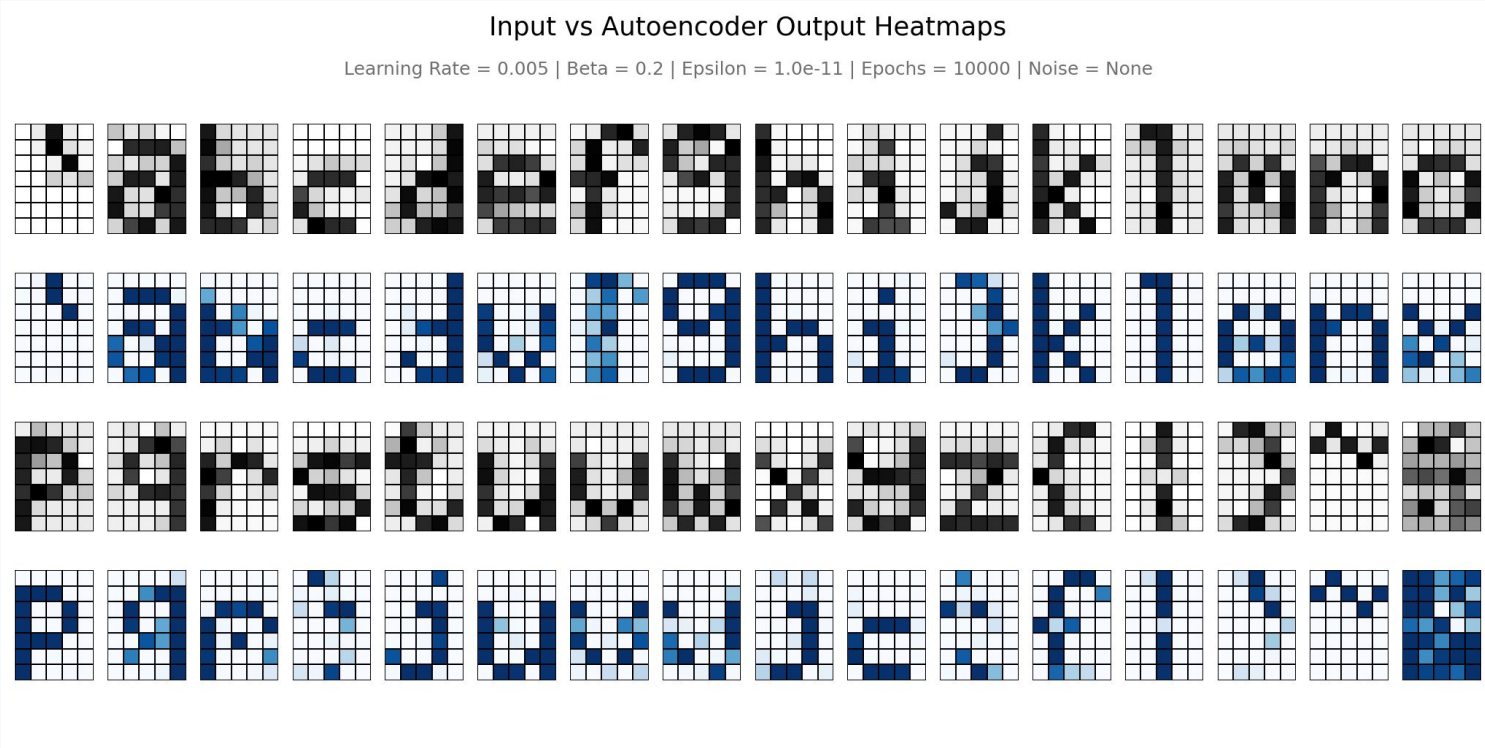
Arquitectura: [60, 50, 30, 10, 5]

Dataset - Ruido 0.1



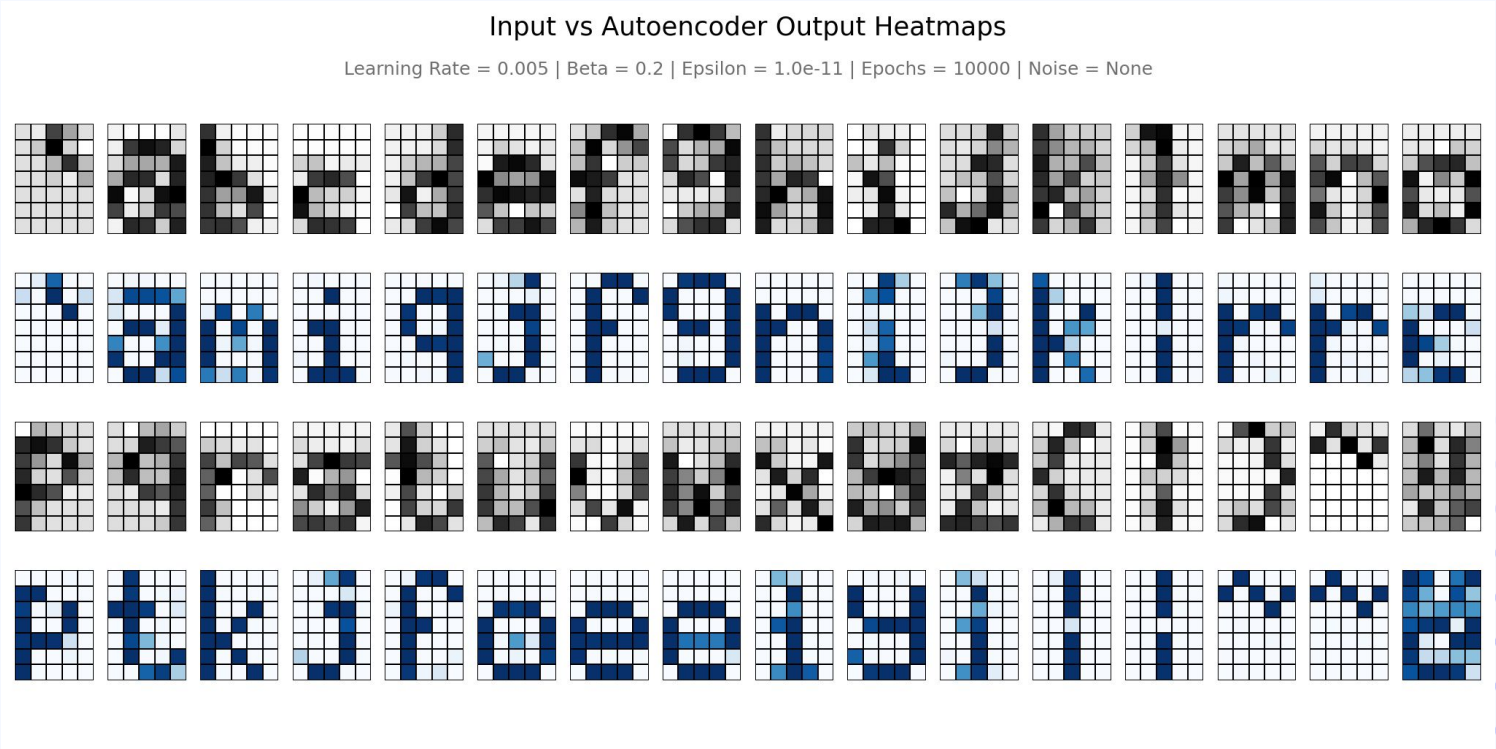
Arquitectura: [60, 50, 40, 30, 20, 10, 5]

Dataset - Ruido 0.3



Arquitectura: [60, 50, 30, 10, 5]

Dataset - Ruido 0.5



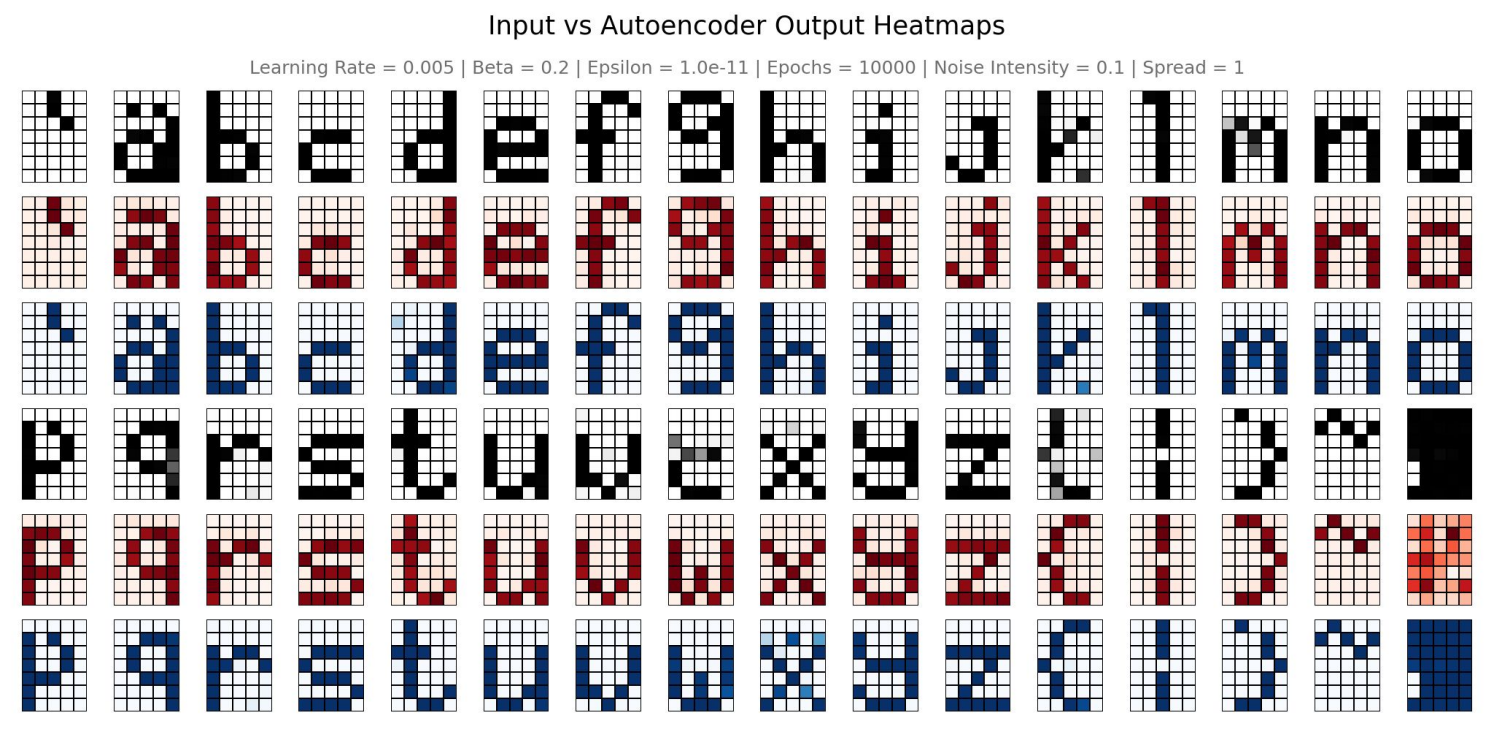
Arquitectura: [60, 50, 30, 10, 5]

Ruido 0.1

Aprendido (paso previo)

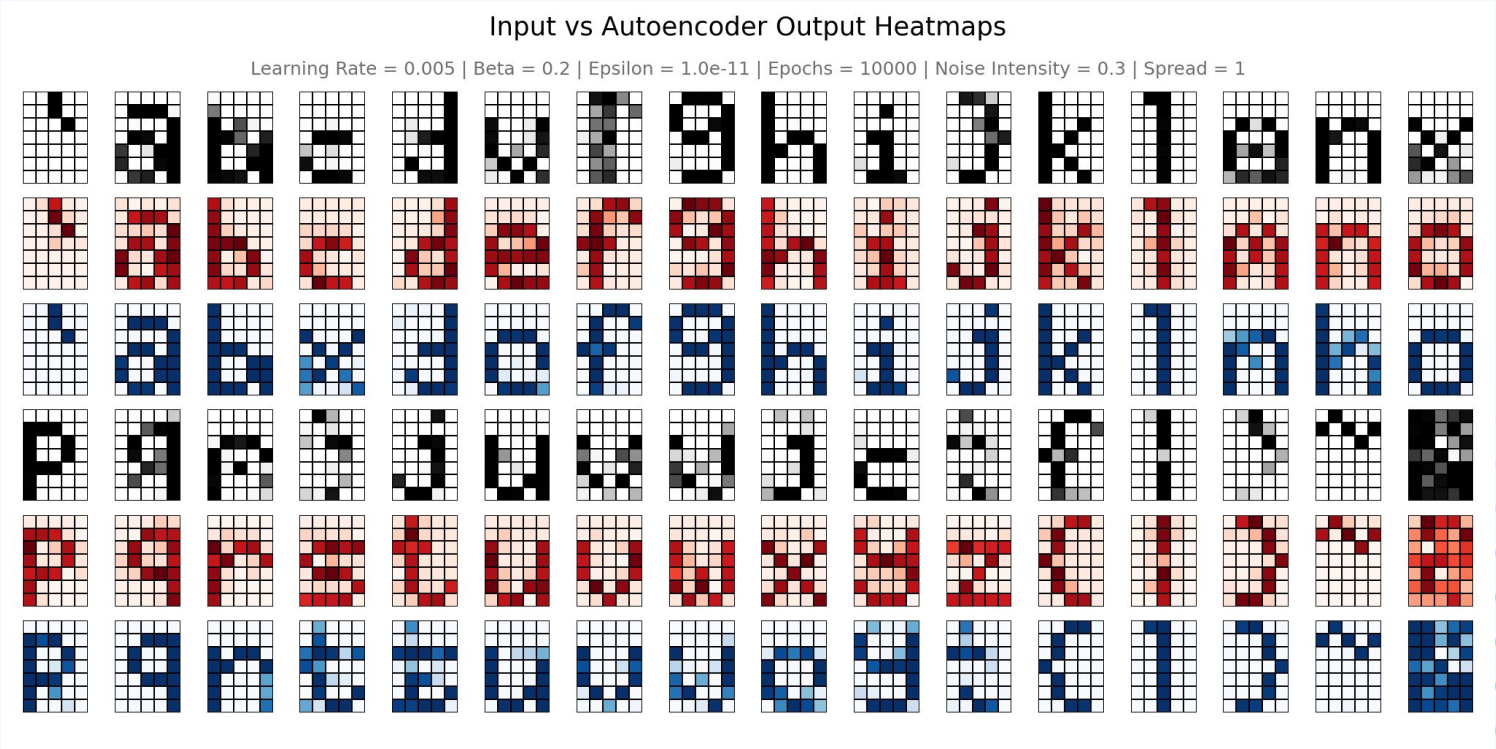
Entrada con ruido

Interpretación



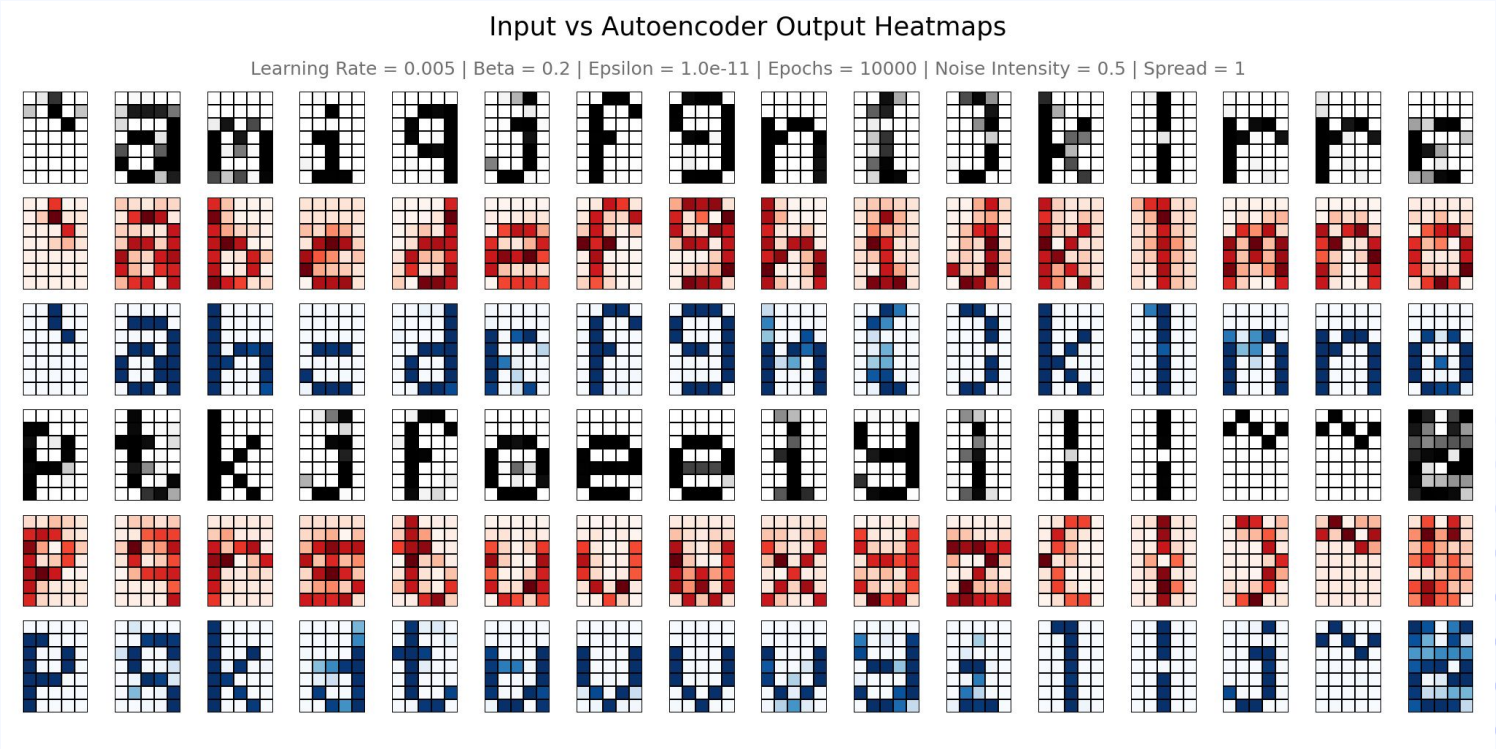
Arquitectura: [60, 50, 30, 10, 5]

Ruido 0.3



Arquitectura: [60, 50, 30, 10, 5]

Ruido 0.5

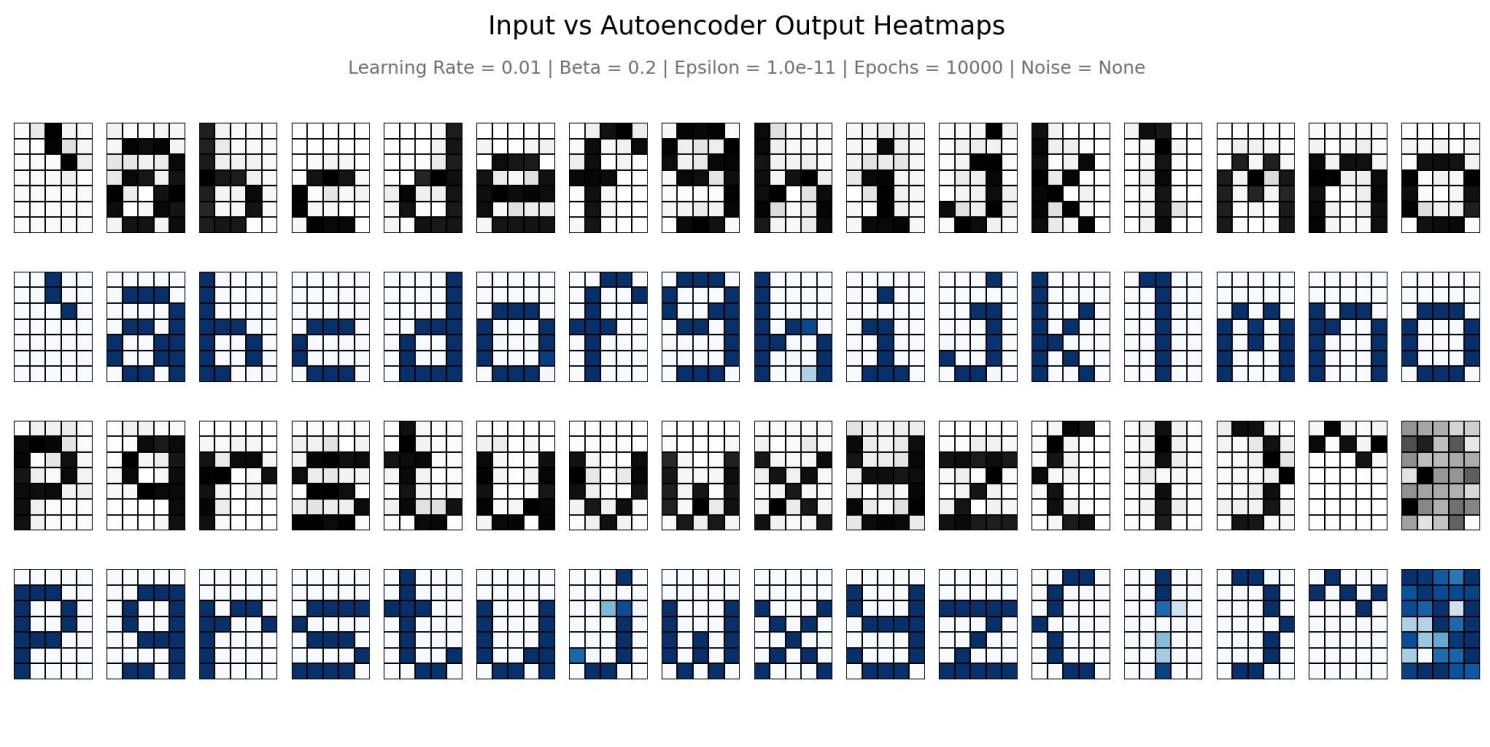


Arquitectura: [60, 50, 30, 10, 5]

Dataset - Ruido 0.1 - LR = 0.01

Entrada con ruido del entrenamiento

Aprendizaje



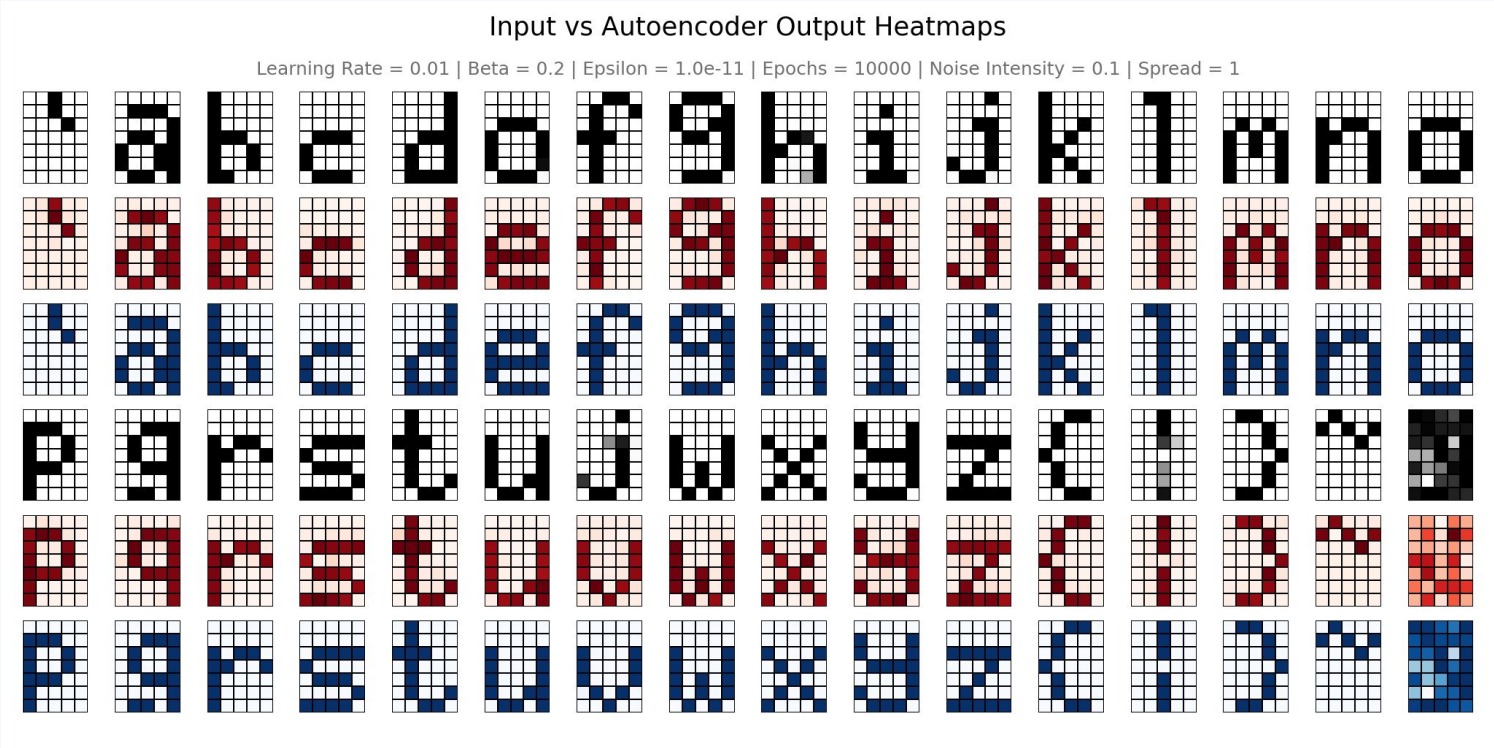
Arquitectura: [60, 50, 30, 10, 5]

Ruido 0.1 - LR = 0.01

Aprendido (paso previo)

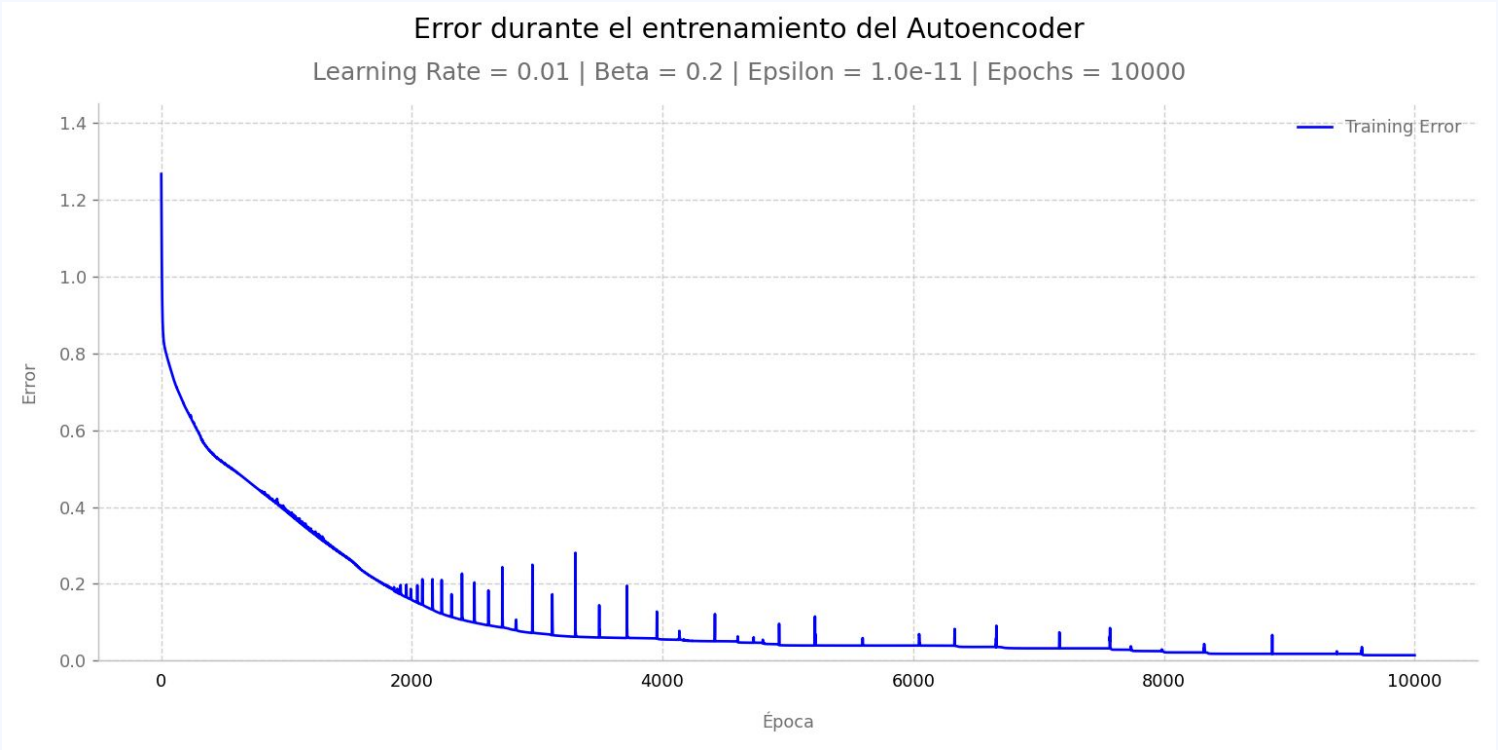
Entrada con ruido

Interpretación



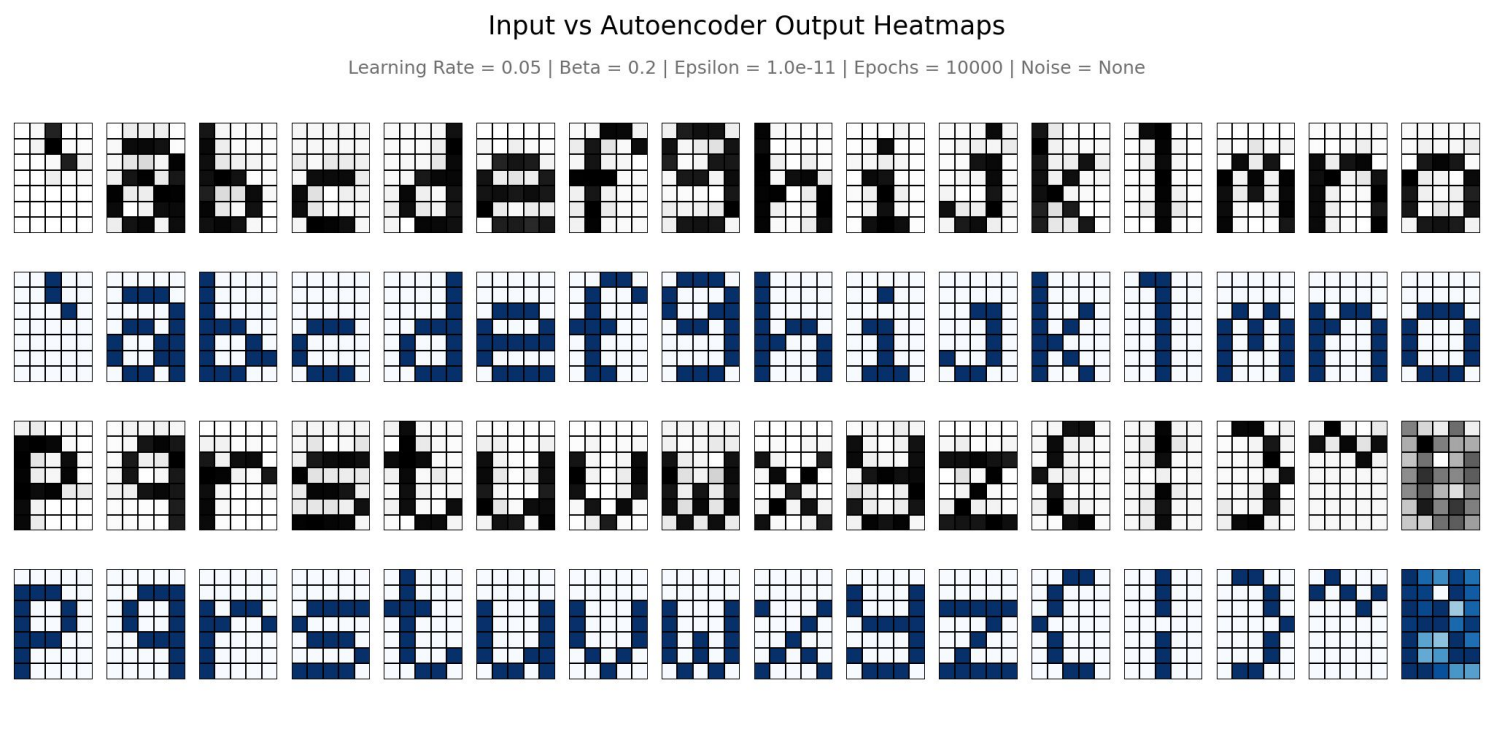
Arquitectura: [60, 50, 30, 10, 5]

Ruido 0.1 - LR = 0.01



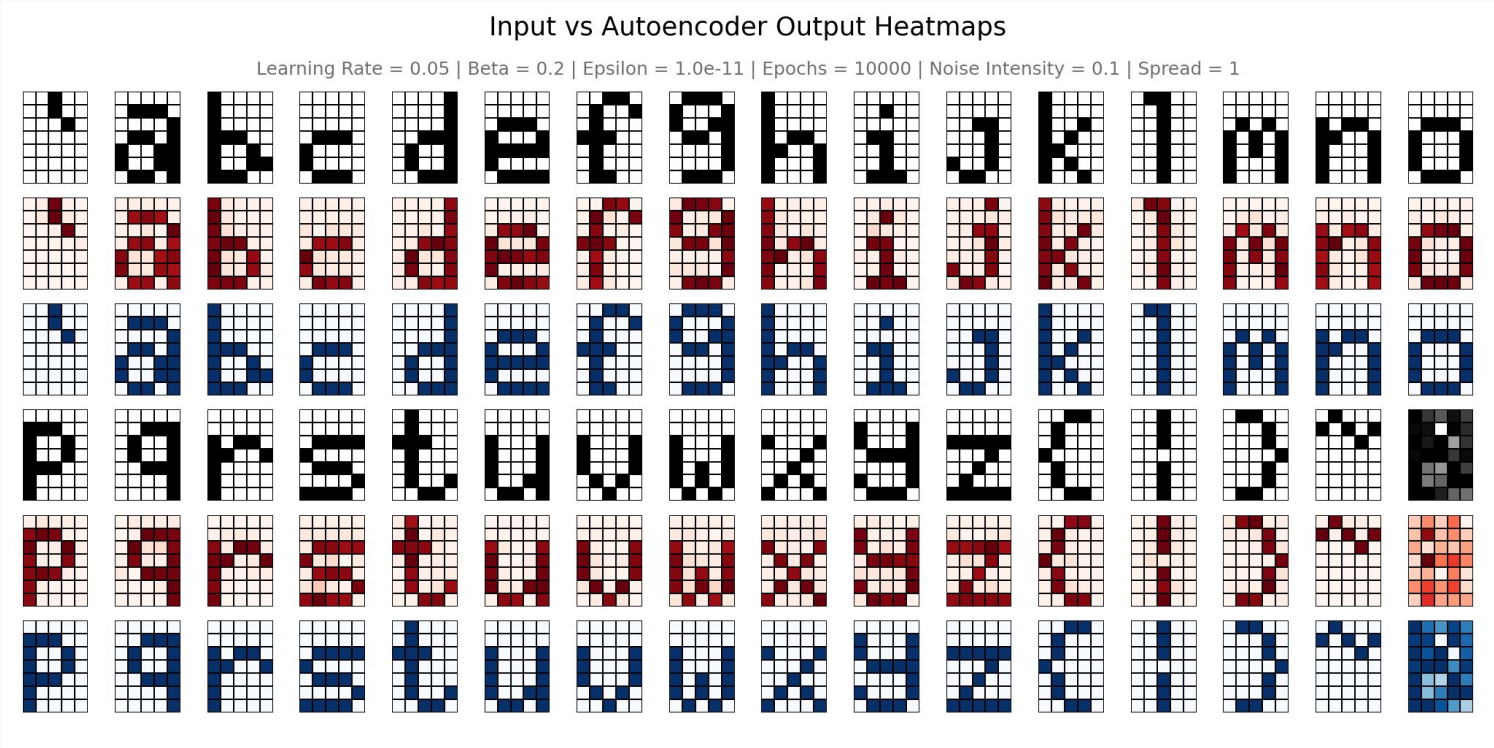
Arquitectura: [60, 50, 30, 10, 5]

Dataset - Ruido 0.1 - LR = 0.05



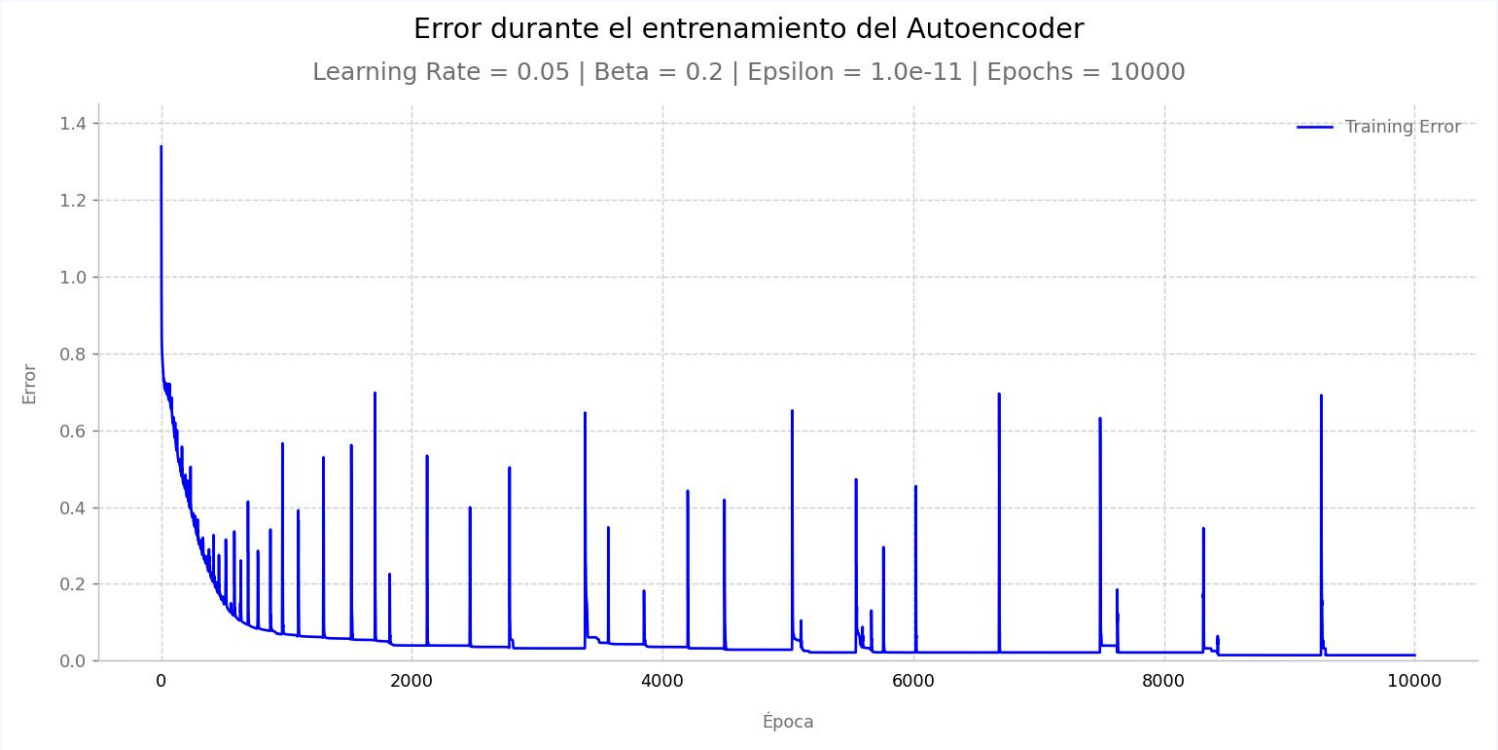
Arquitectura: [60, 50, 30, 10, 5]

Ruido 0.1 - LR = 0.05



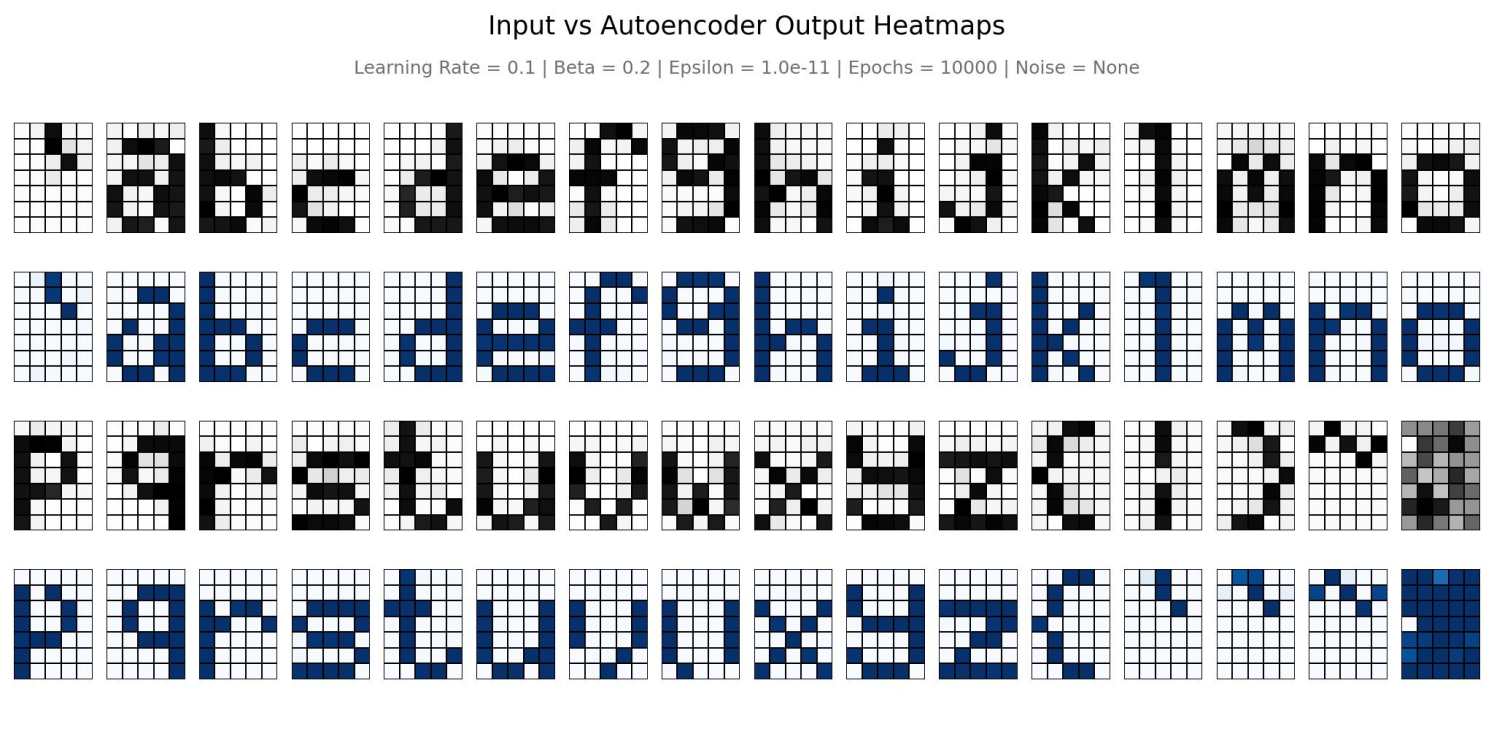
Arquitectura: [60, 50, 30, 10, 5]

Ruido 0.1 - LR = 0.05



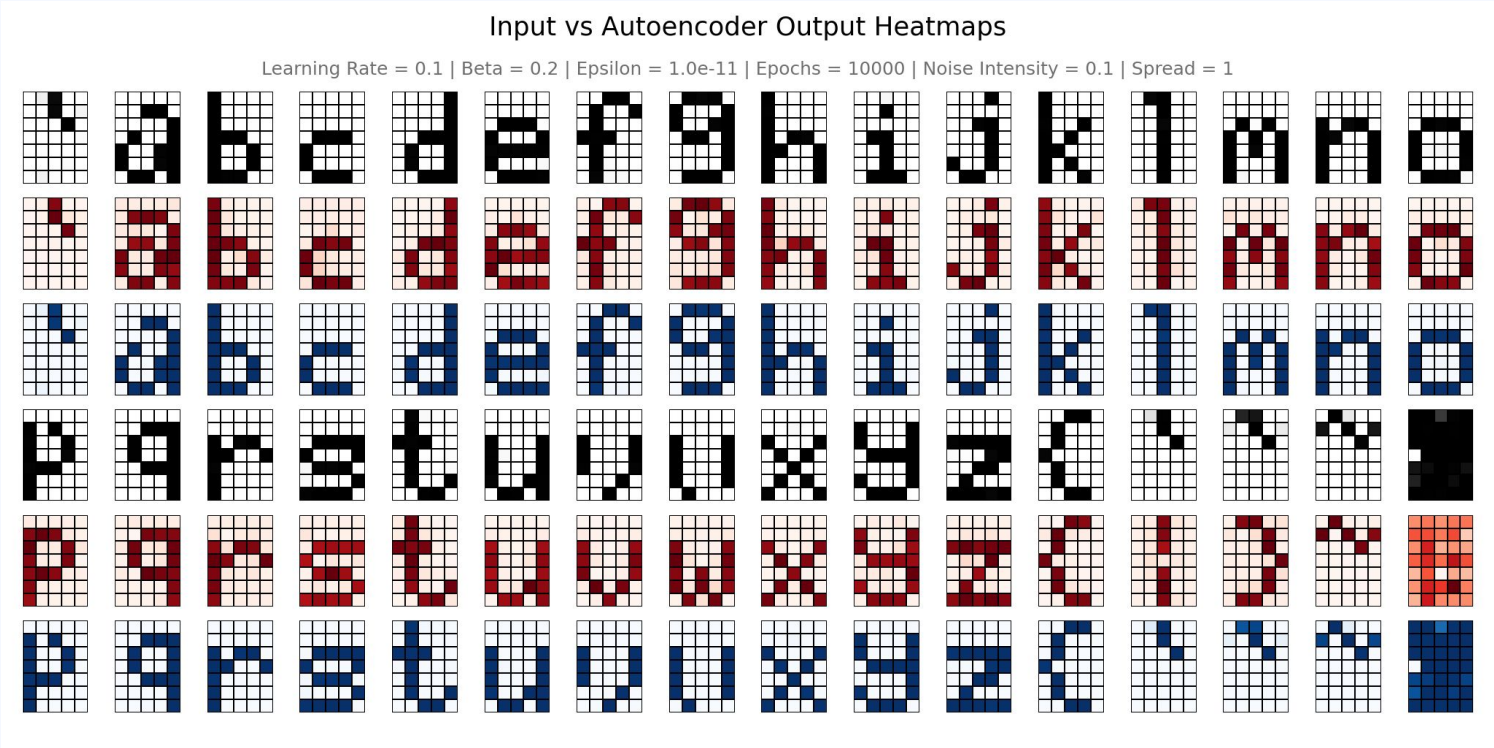
Arquitectura: [60, 50, 30, 10, 5]

Dataset - Ruido 0.1 - LR = 0.1



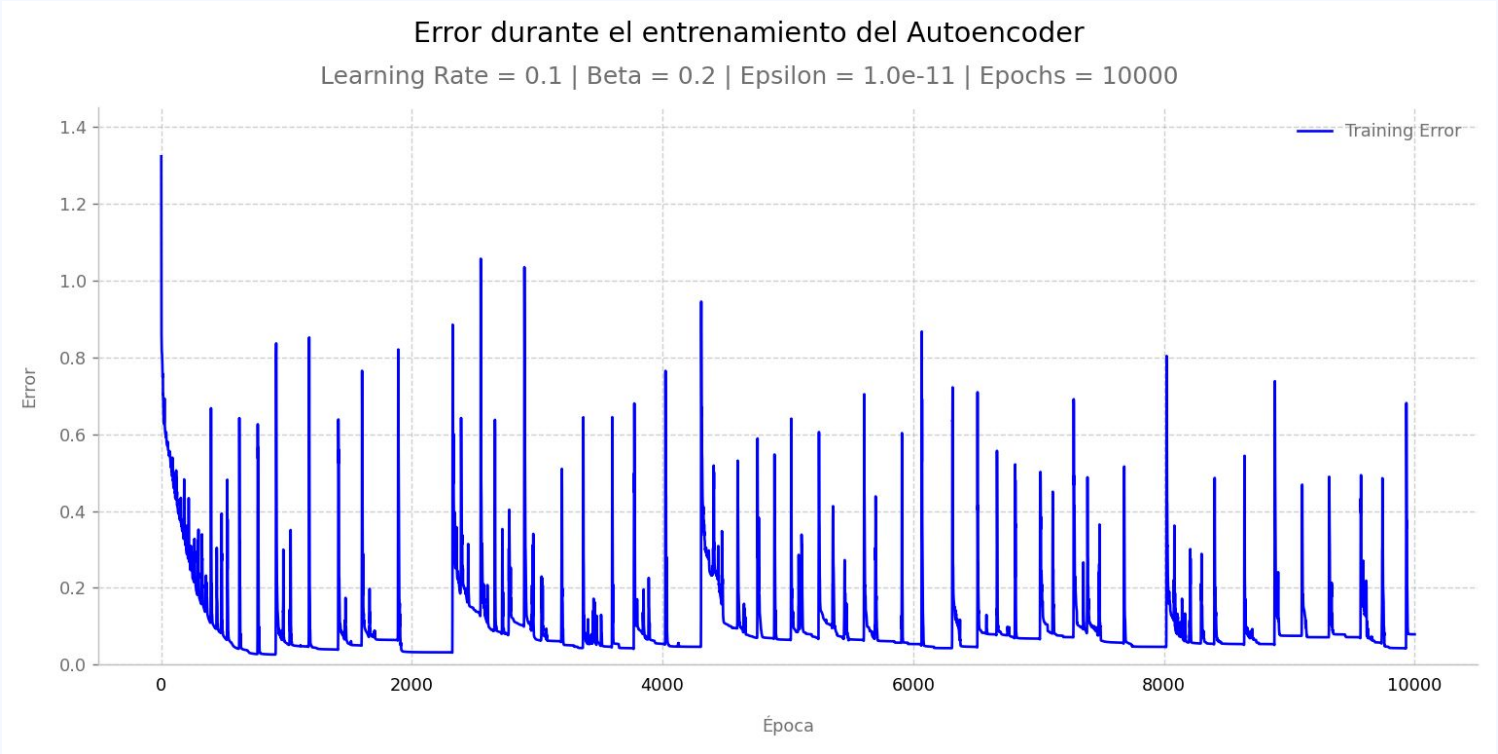
Arquitectura: [60, 50, 30, 10, 5]

Ruido 0.1 - LR = 0.1



Arquitectura: [60, 50, 30, 10, 5]

Ruido 0.1 - LR = 0.1



Arquitectura: [60, 50, 30, 10, 5]

Conclusiones

- Los autoencoders pueden detectar y filtrar muy bien el ruido
- Cambiar la arquitectura, learning rate y cantidad de épocas altera significativamente el resultado y tiempos de ejecución
- ¡Cuidado con el error!

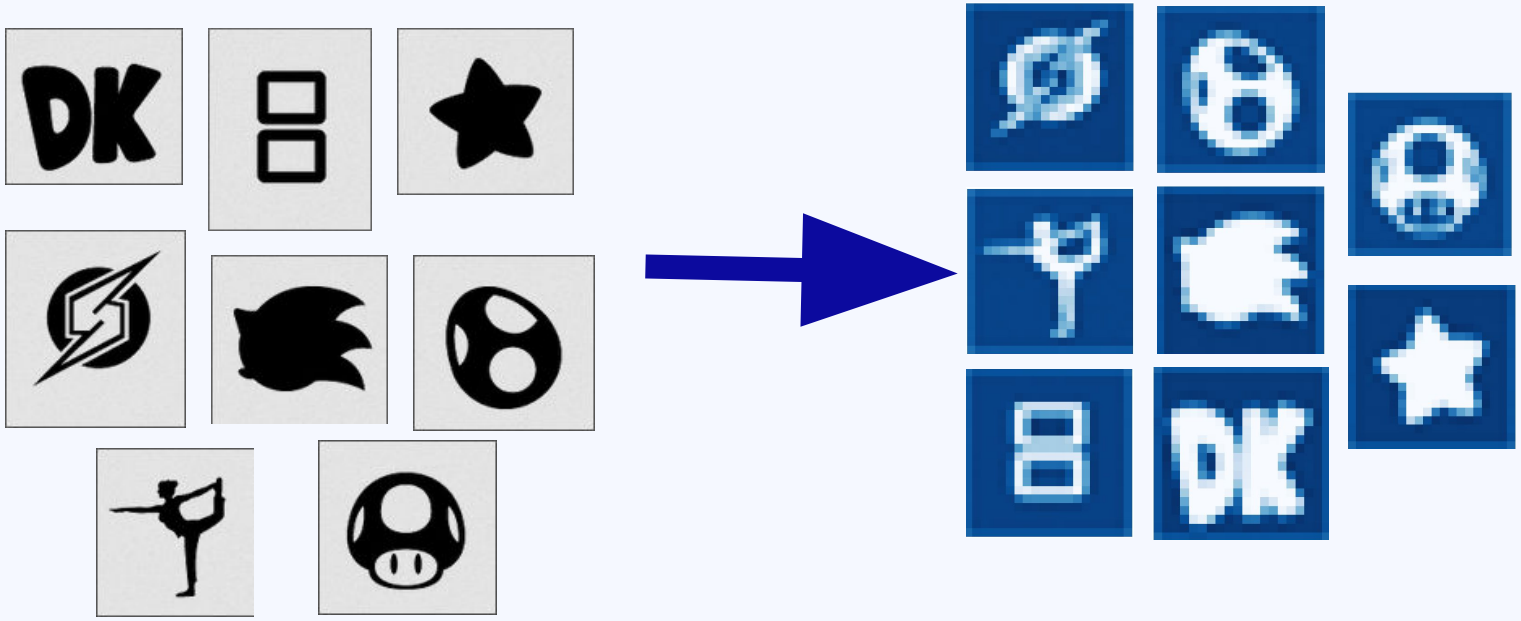
03
Autoencoder Variacional

Conjunto de datos nuevo

Para este conjunto de datos elegimos los logotipos de alguna de las franquicias de Nintendo

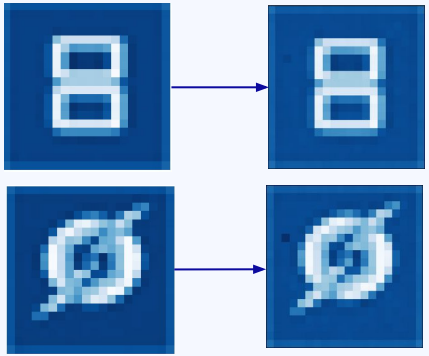
Conjunto de datos nuevo

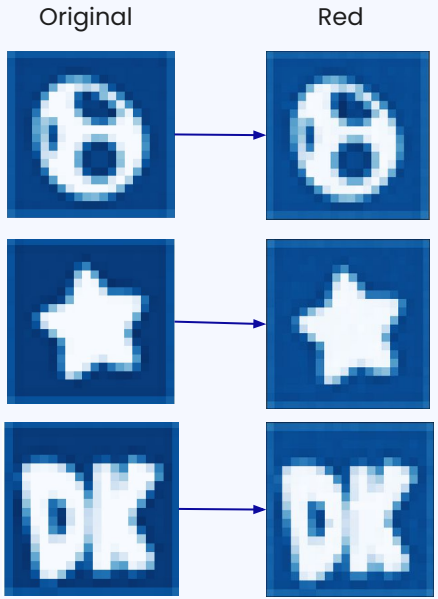
Para poder entrenar a la red, procesamos las imágenes para que tengan un mismo tamaño y resolución, y los cambiamos a una escala de azules

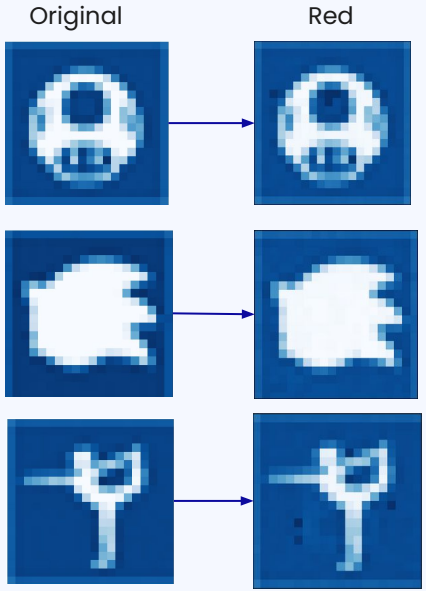


Algunos Resultados

Arquitectura: **400-300-200-100-50-2-50-100-200-300-400**
Learning Rate: 0.005
Epochs: 1500
Optimizer: Adam







Generación de Nuevos Logos

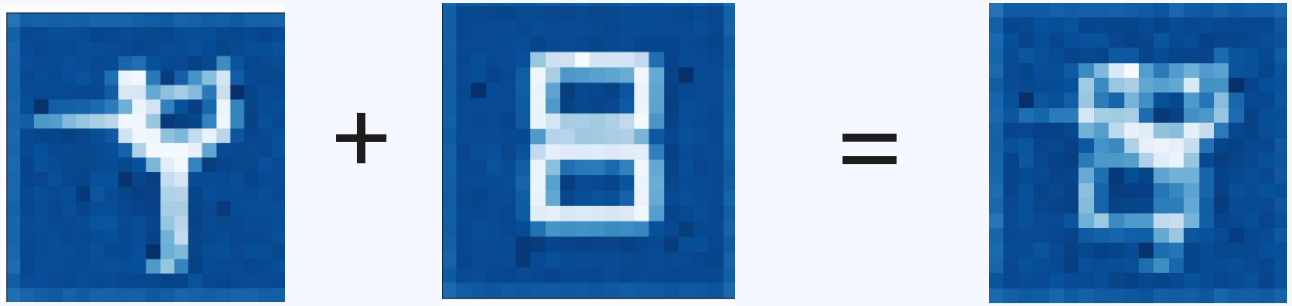
Para poder generar nuevos logos, codificamos las imágenes y generamos sus valores en el espacio latente.

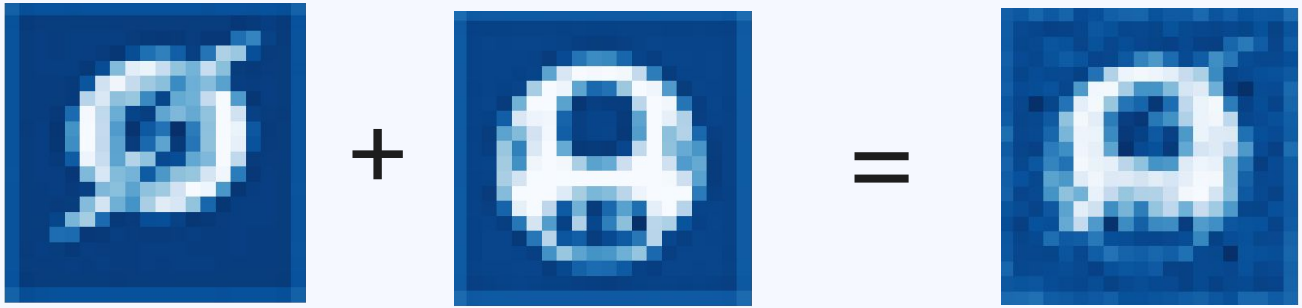
```
latent_space_1 = autoencoder.encode(img1)
latent_space_1 = autoencoder.encode(img1)
```

Luego, decodificamos la imagen resultante del promedio de sus valores:

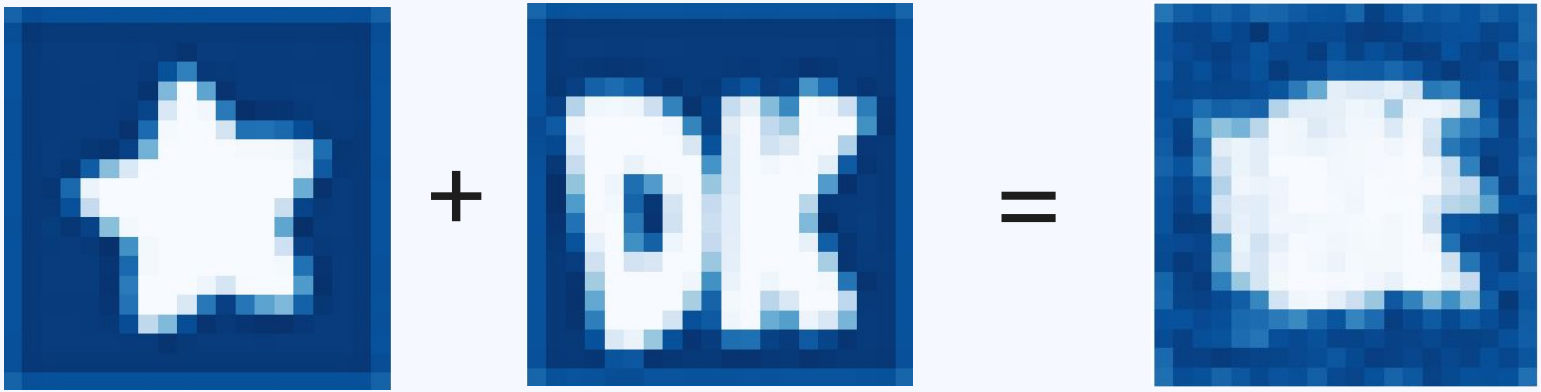
```
mean_latent_space = 0.5 * (latent_space_1 + latent_space_2)
    new_img = autoencoder.decode(mean_latent_space)
```

Generación de Nuevos Logos





Generación de Nuevos Logos



Conclusiones

- Los VAEs permiten generar nuevas muestras y realizar interpolaciones más suaves sobre el espacio latente.
- La representación del espacio latente en un autoencoder no variacional no tiene una estructura probabilística, lo que limita la capacidad de generar nuevas muestras.

Muchas gracias!