Trabajo Práctico N°5

Deep Learning

Grupo 1

- Augusto Henestrosa
- Francisco Choi
- Nicolás de la Torre

Desarrollo

Python 3.8.0

Para Autoencoder común y DAE:

- Implementación 100% propia
- Utilizando solo numpy y matplotlib

Para Autoencoder variacional:

- Keras
- Tensorflow

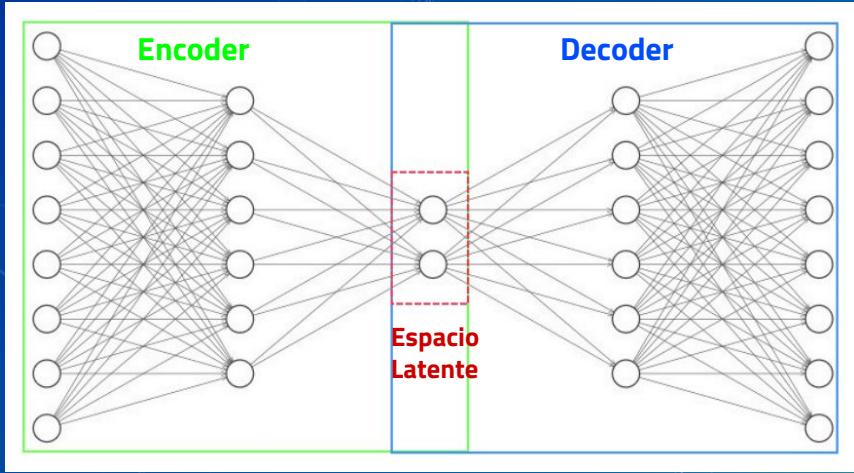
Introducción

¿Qué es un Autoencoder?

- Redes neuronales
- Comprimen la entrada en una representación de espacio latente y luego reconstruyen la salida de esta representación.

¿Para qué sirven?

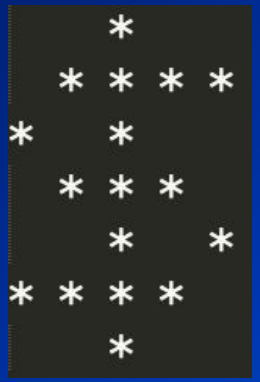
- Reducción de dimensionalidad
- Identificación de anomalías
- Eliminación de ruido
- Generación de nuevos datos



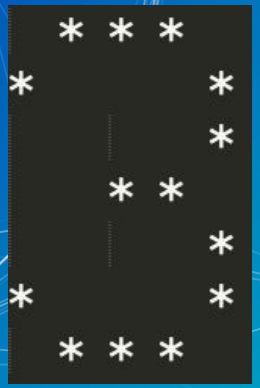
Dataset

Dataset - Símbolos

```
[0x04, 0x0f, 0x14, 0x0e, 0x05, 0x1e, 0x04]
```



```
[0x0e, 0x11, 0x01, 0x06, 0x01, 0x11, 0x0e]
```



Autoencoder Lineal

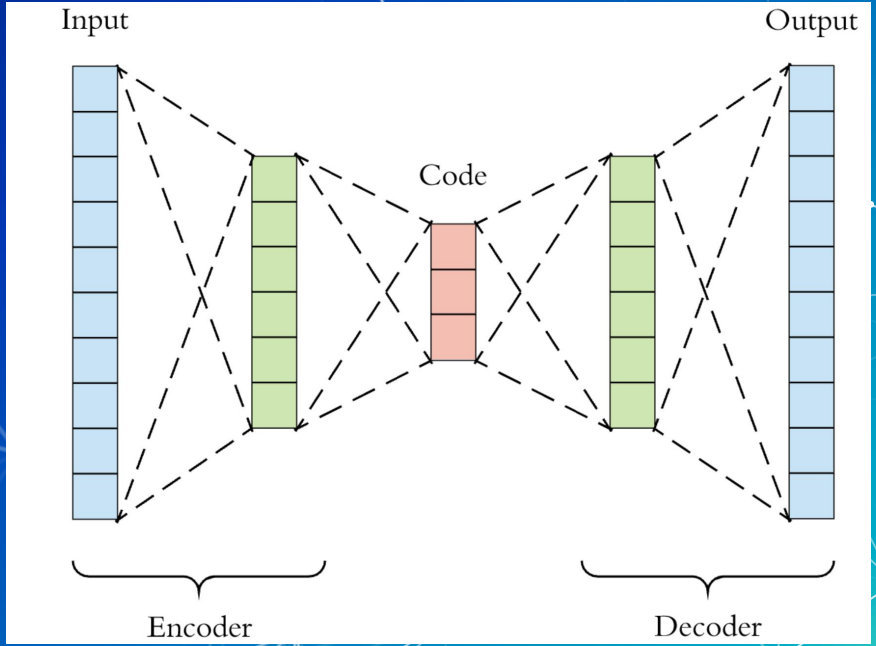
Autoencoder Lineal - Estructuras

Configuración

- Conjunto de entrenamiento: 4 letras
- Épocas máximas: 500
- Error mínimo: 1
- Uso de optimizaciones: Momentum + Costo entrópico + Adaptativo

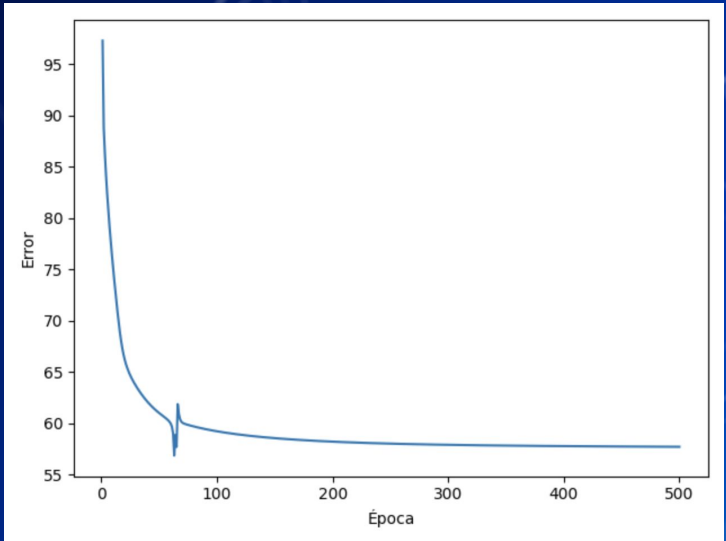
Estructuras usadas:

- [35,2,35]
- [35,3,2,3,35]
- [35,5,3,2,3,5,35]
- [35,16,2,16,35]



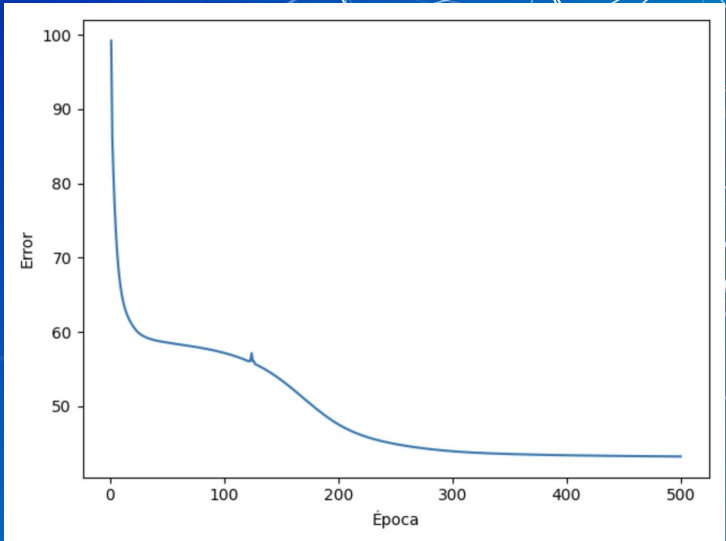
Estructuras

Estructura: [35,5,3,2,3,5,35]



Final error: 57,35

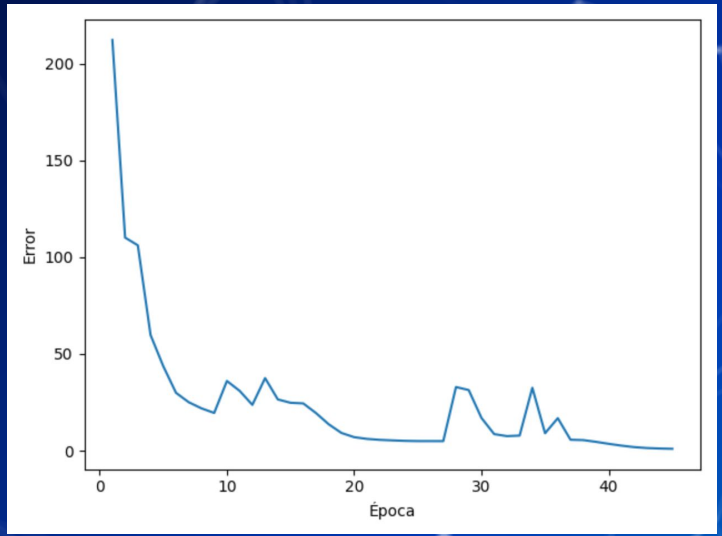
Estructura: [35,3,2,3,35]



Final error: 41,22

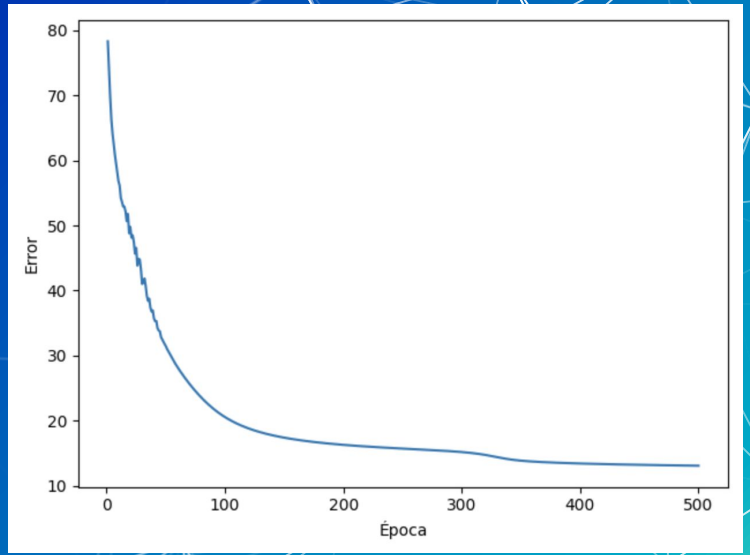
Estructuras

Estructura: [35,16,2,16,35]



Final error: 0,90

Estructura: [35,2,35]



Final error: 13,06

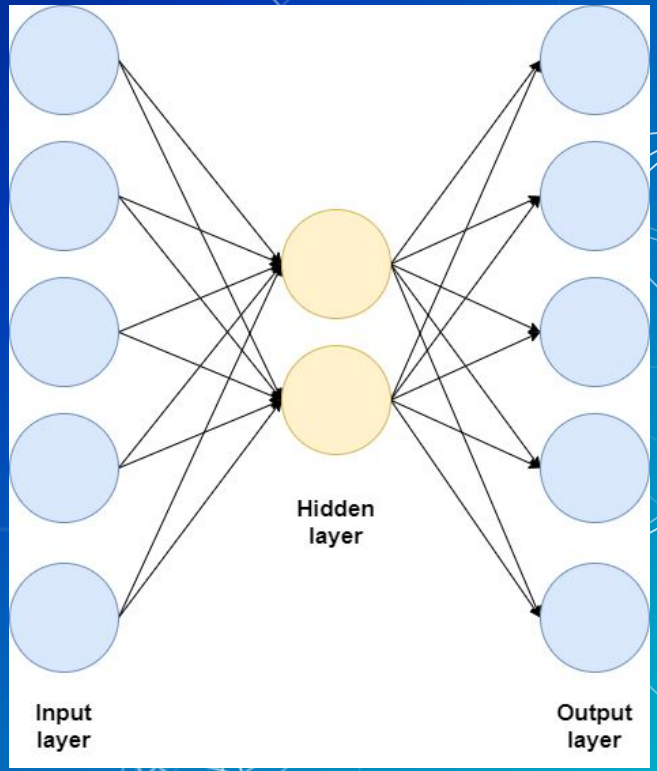
Autoencoder Lineal - Optimizaciones

Configuración

- Conjunto de entrenamiento: 4 letras
- Épocas máximas: 500
- Error mínimo: 1
- Estructura: [35,16,2,16,32]

Actualización de pesos:

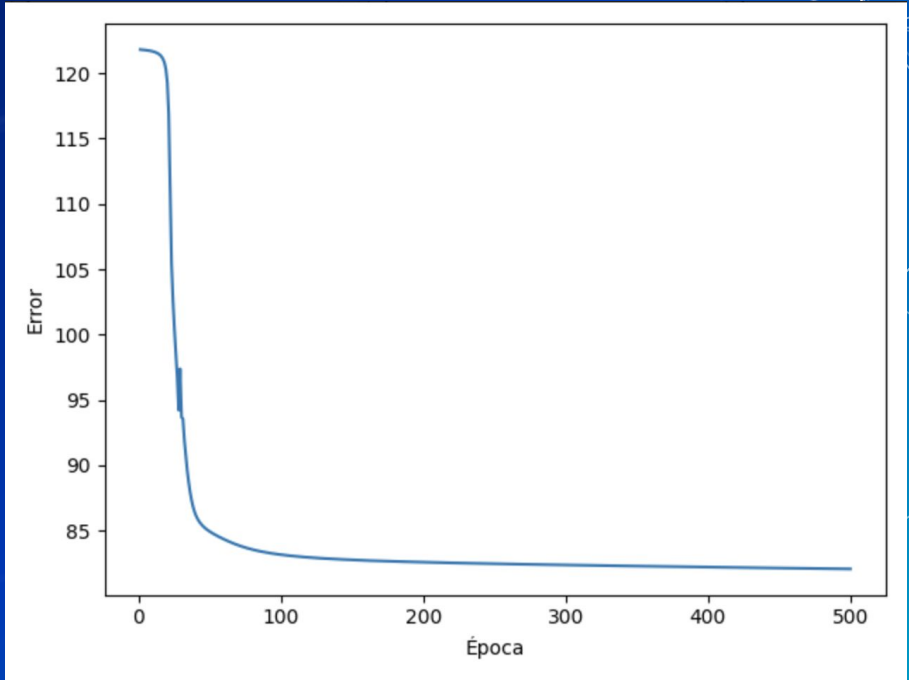
- Actualización sin optimización
- Costo entrópico
- Momentum (0,9)
- Adaptativo:
  - a= 10^{-4}, b=10^{-5}



[16, 2, 16] hidden layers

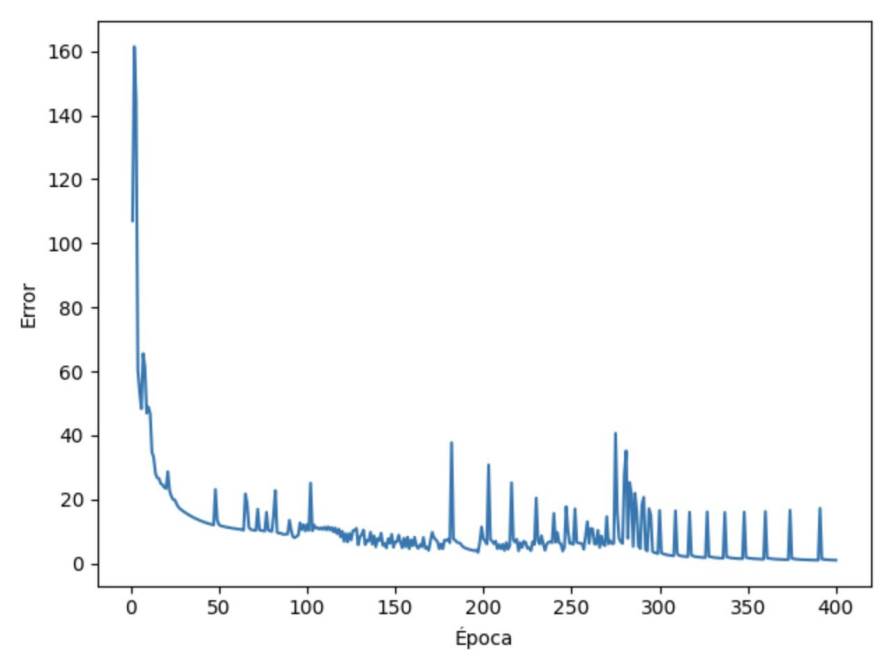
Actualización de pesos

Actualización sin optimización



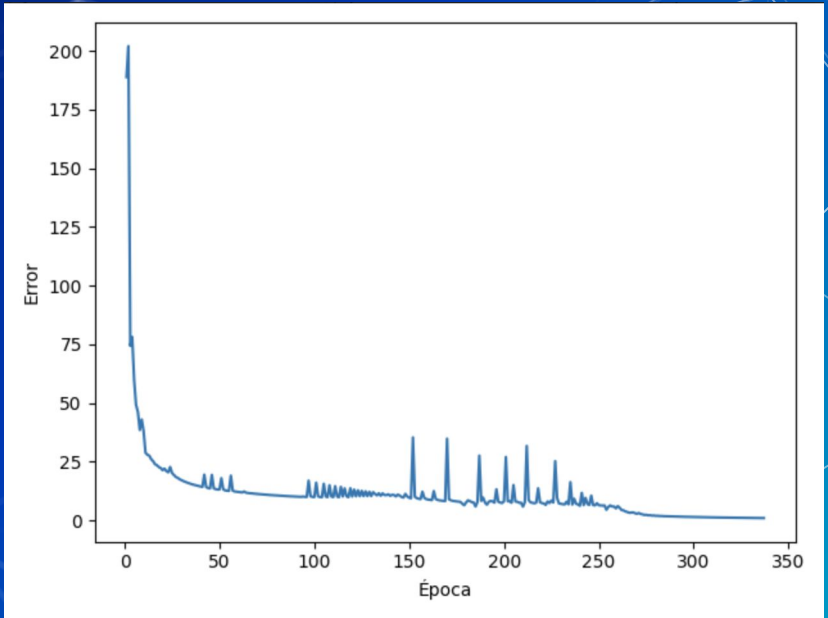
Actualización de pesos

Costo entrópico



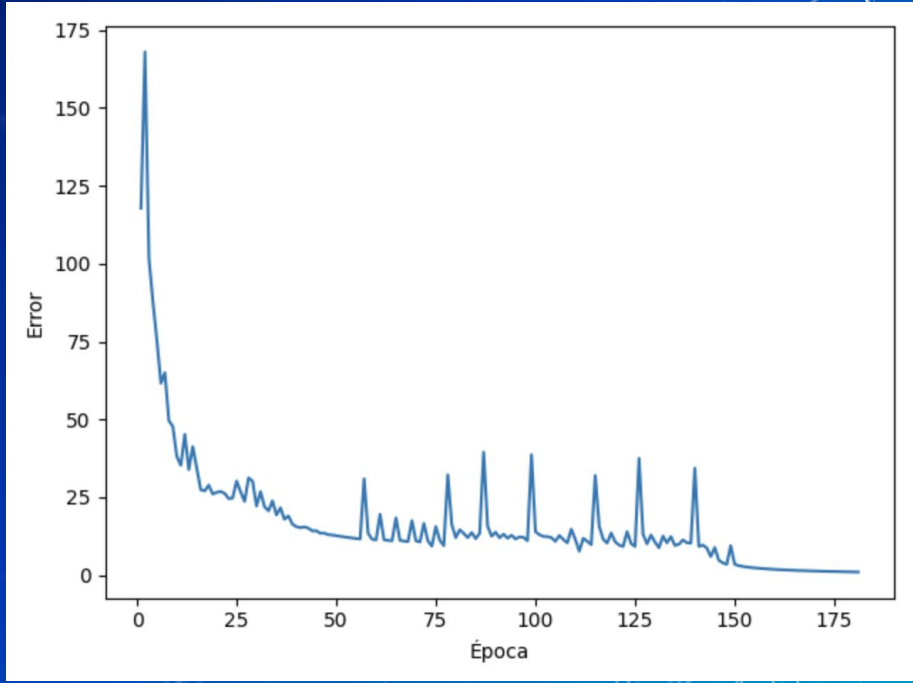
Actualización de pesos

Costo entrópico + Momentum



Actualización de pesos

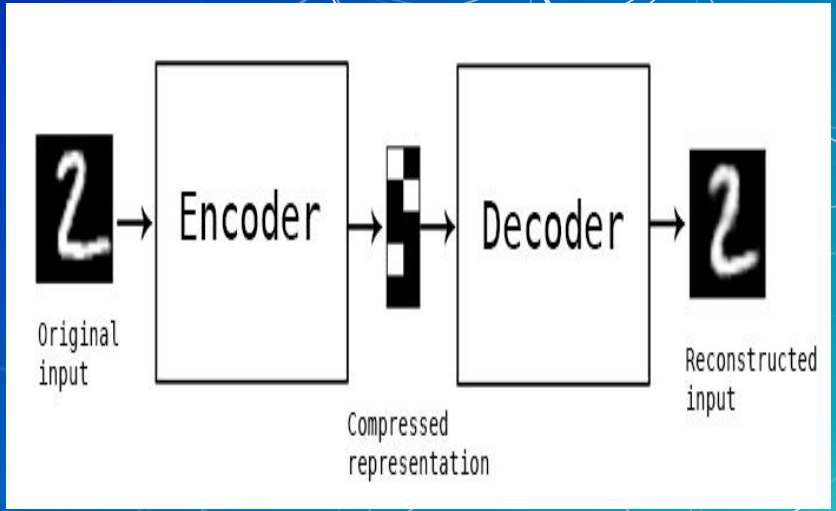
Costo entrópico + Momentum + Adaptativo



Autoencoder Lineal - Representación 2D + Generación desde espacio latente

Configuración

- Conjunto de entrenamiento: 4 letras
- Épocas máximas: 500
- Error mínimo: 1
- Estructura: [35,16,2,16,32]
- Uso de optimizaciones: Momentum + Costo entrópico + Adaptativo



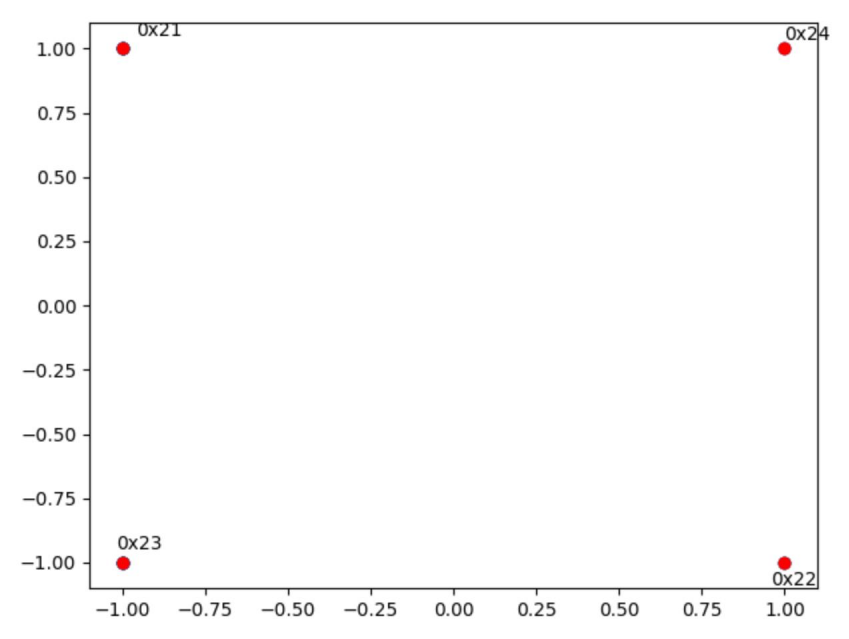
Variación del porcentaje del sample utilizado:

- 4 letras
- 8 letras
- 16 letras
- 24 letras

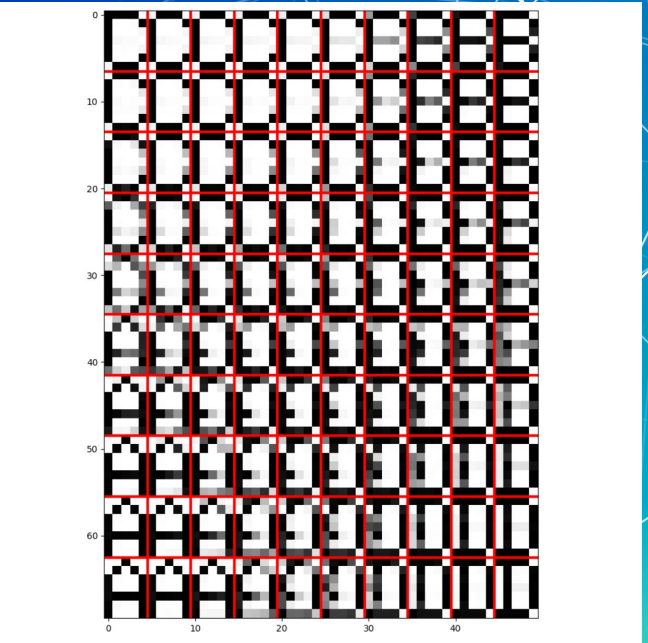
Representación 2D + Generación desde espacio latente

Conjunto de entrenamiento: 4 letras

Espacio latente representación 2D



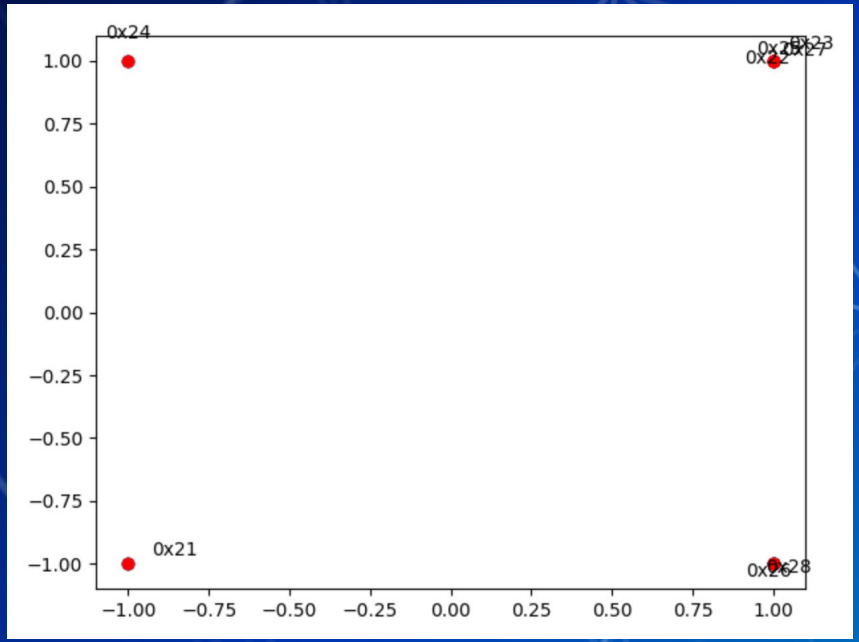
Generación desde espacio latente



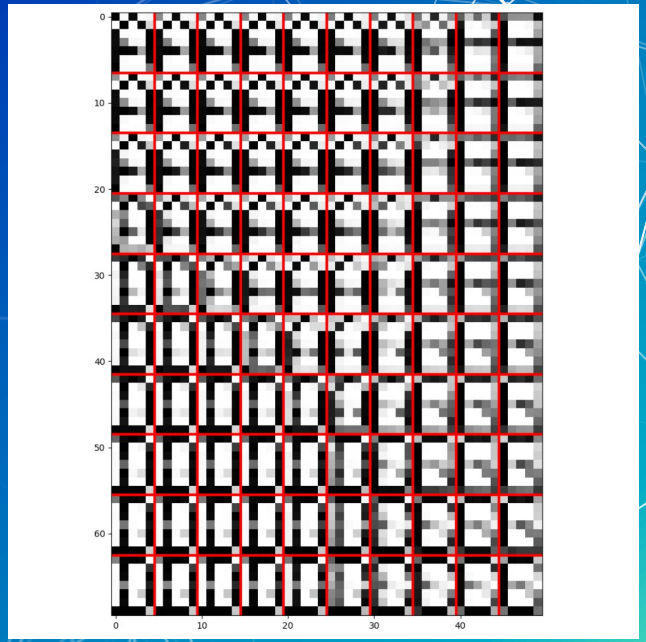
Representación 2D + Generación desde espacio latente

Conjunto de entrenamiento: 8 letras

Espacio latente representación 2D



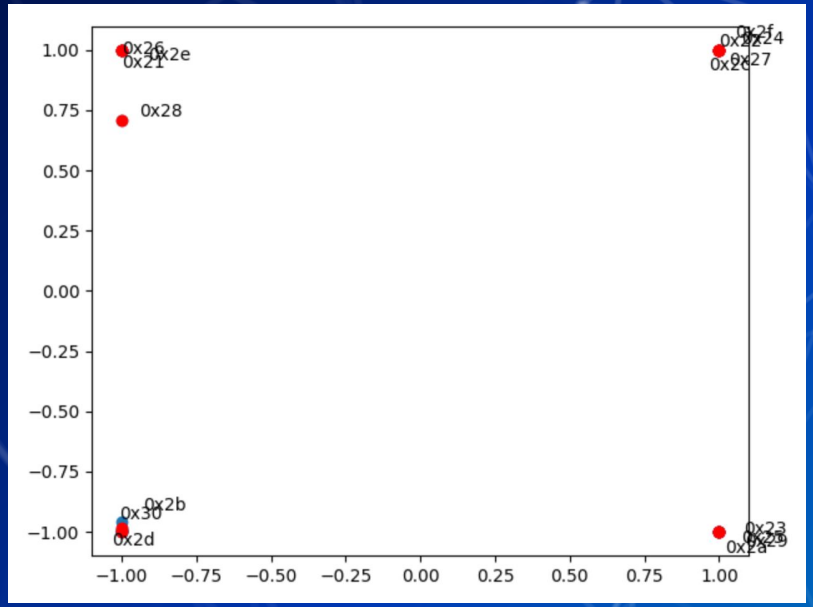
Generación desde espacio latente



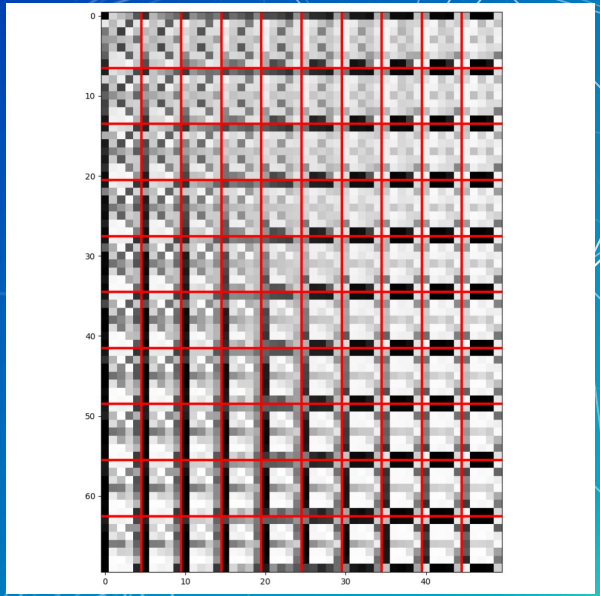
Representación 2D + Generación desde espacio latente

Conjunto de entrenamiento: 16 letras

Espacio latente representación 2D



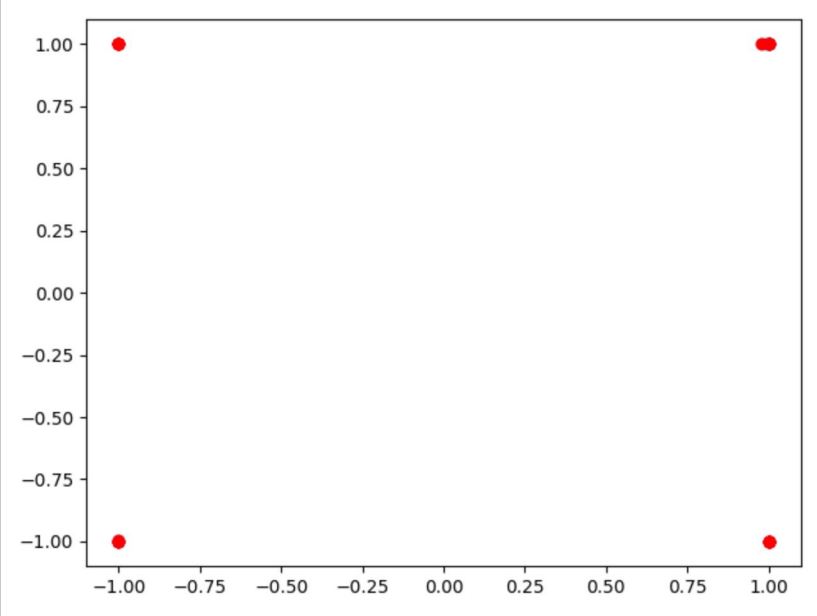
Generación desde espacio latente



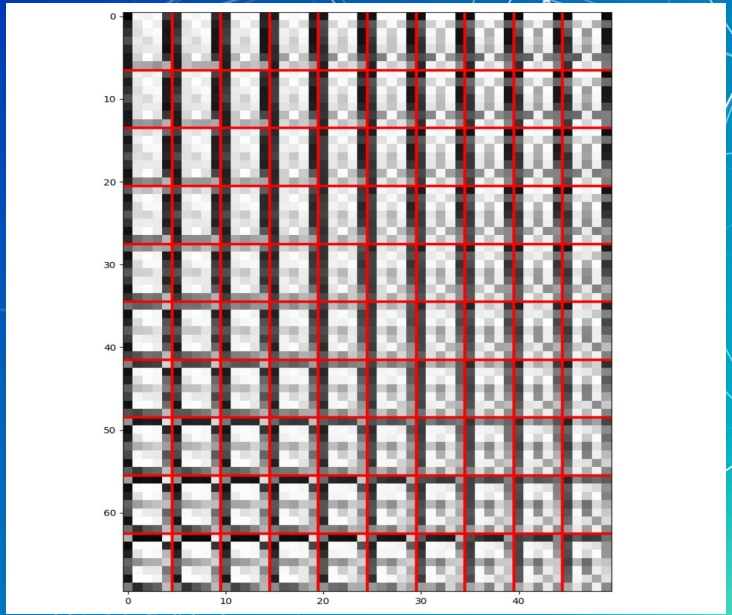
Representación 2D + Generación desde espacio latente

Conjunto de entrenamiento: 24 letras

Espacio latente representación 2D

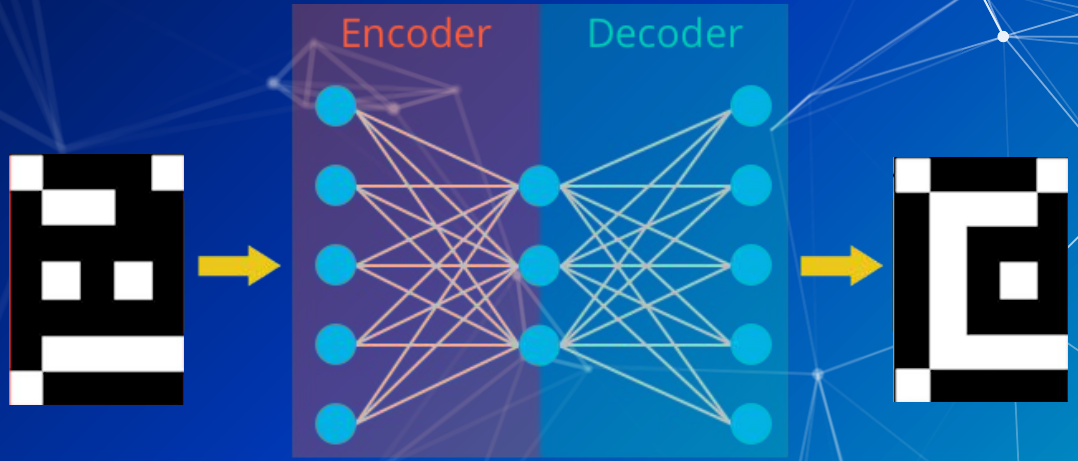


Generación desde espacio latente



Denoising Autoencoder

Denoising Autoencoder



Denoising Autoencoder - Pruebas

Perceptrón Multicapa

- Aprendizaje: 0.01
- Función de costo: entrópica
- Función de activación: tanh
- Eta adaptativo
- Momentum
- 35 entradas; 35 salidas
- 1 capa intermedia (12 nodos)

Parametros variables

- Dataset
  - 25%
  - 50%
- Mutación con probabilidad
  - 0.05
  - 0.10
  - 0.15

Denoising Autoencoder - Pruebas

Entrenamiento

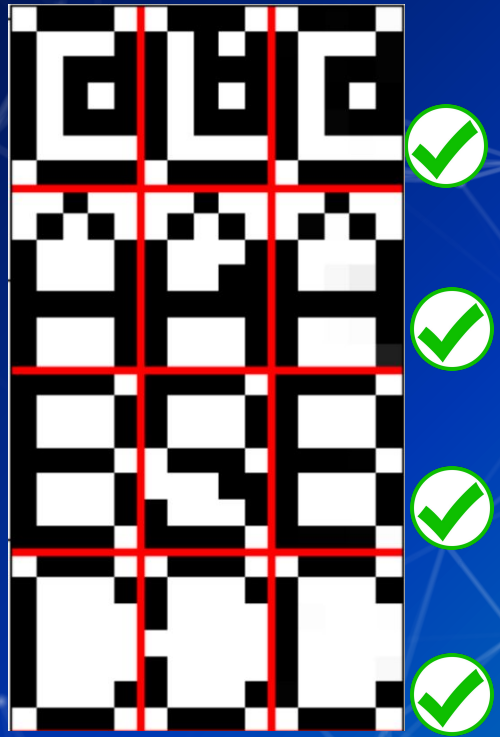
- Ruido salt & pepper
  - Modificación con probabilidad de un pixel
- Para cada simbolo:
  - 3 variaciones

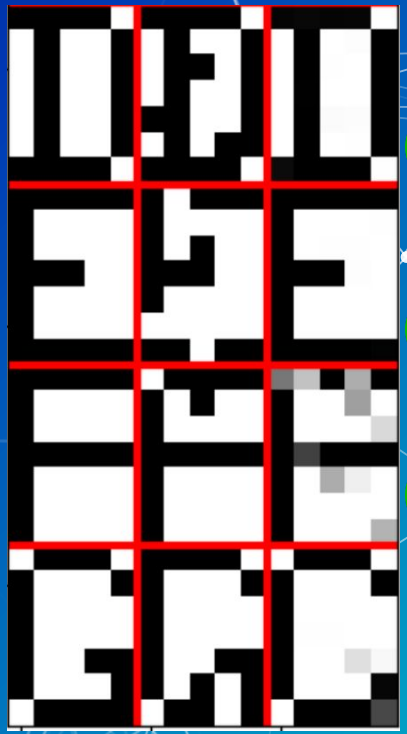
Testeo

- Se genera una nueva variación que no pertenece al conjunto de training
- Se contrasta para ver si es posible identificarla con nuestro criterio

Denoising Autoencoder - Resultados

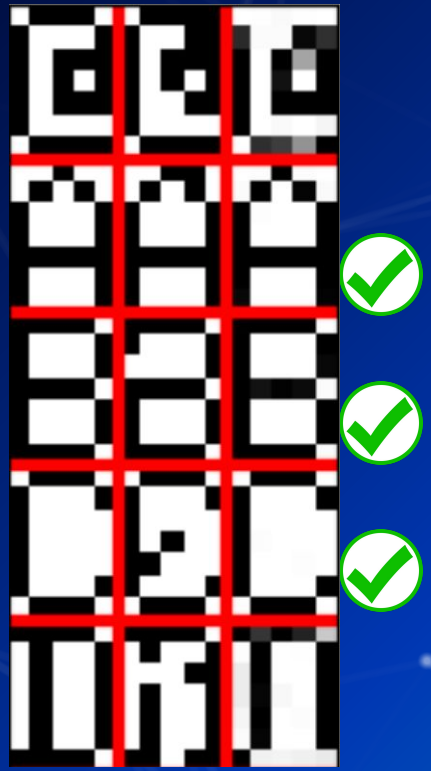
25% del conjunto de entrenamiento - Probabilidad de ruido: 5%

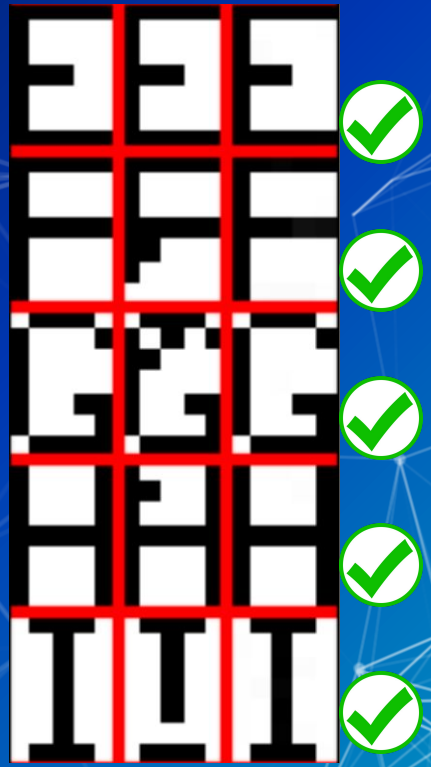


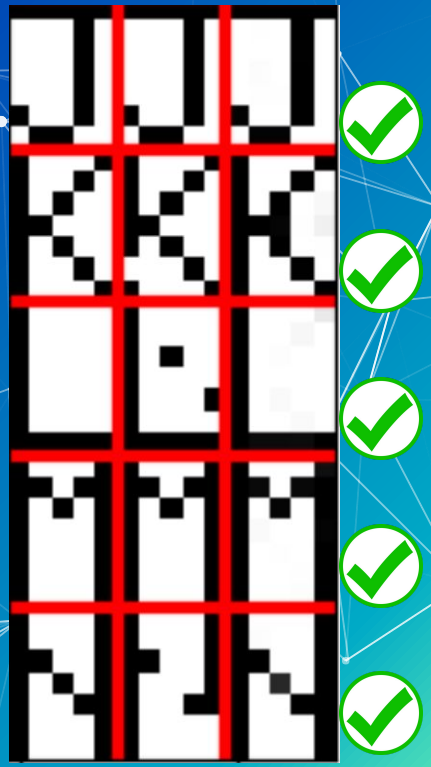


Denoising Autoencoder - Resultados

50% del conjunto de entrenamiento - Probabilidad de ruido: 5%

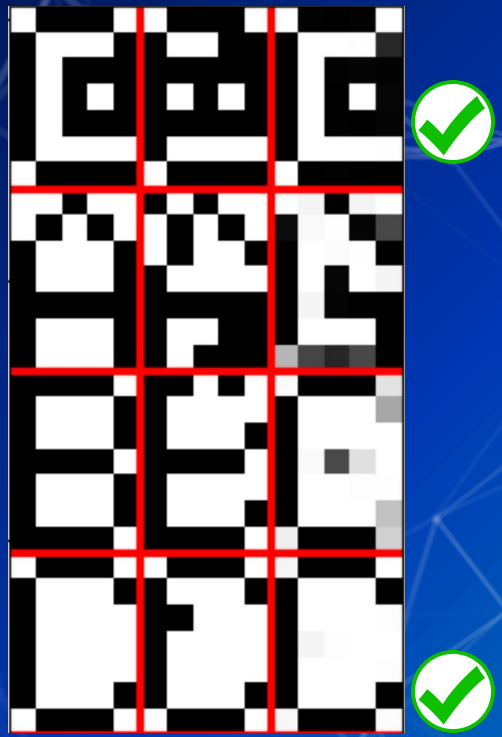


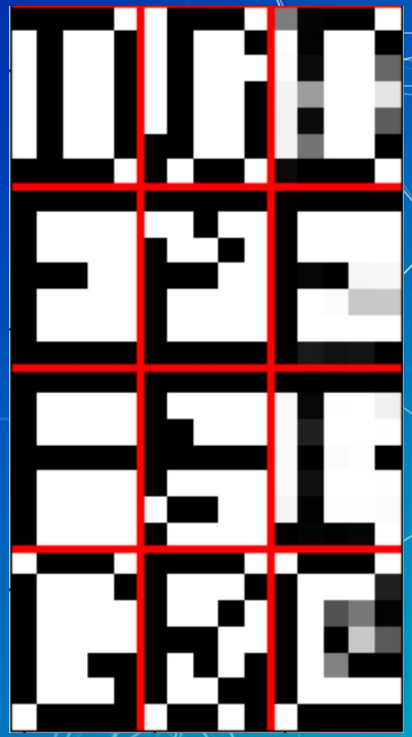




Denoising Autoencoder - Resultados

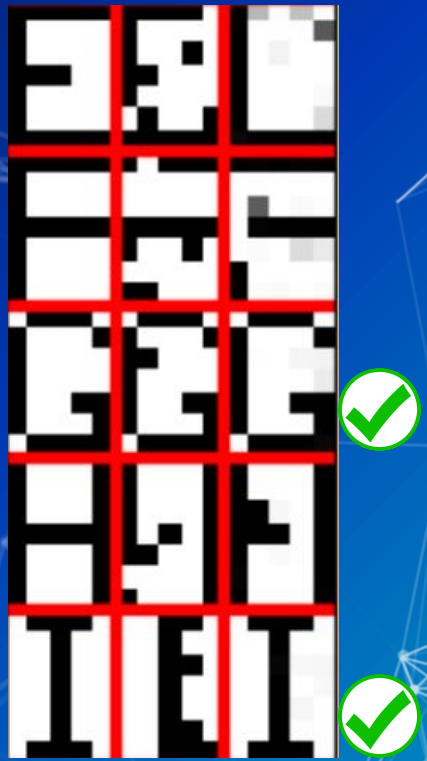
25% del conjunto de entrenamiento - Probabilidad de ruido: 10%

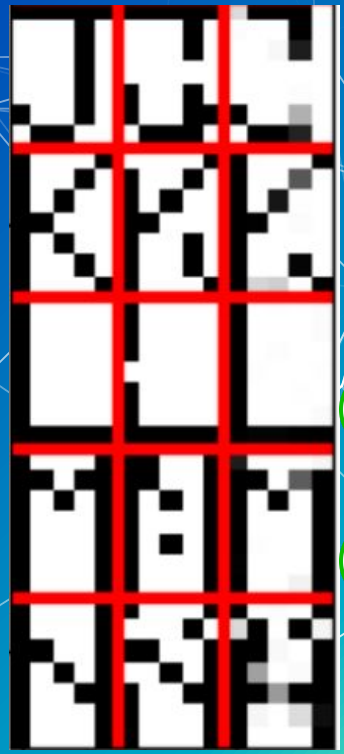




Denoising Autoencoder - Resultados

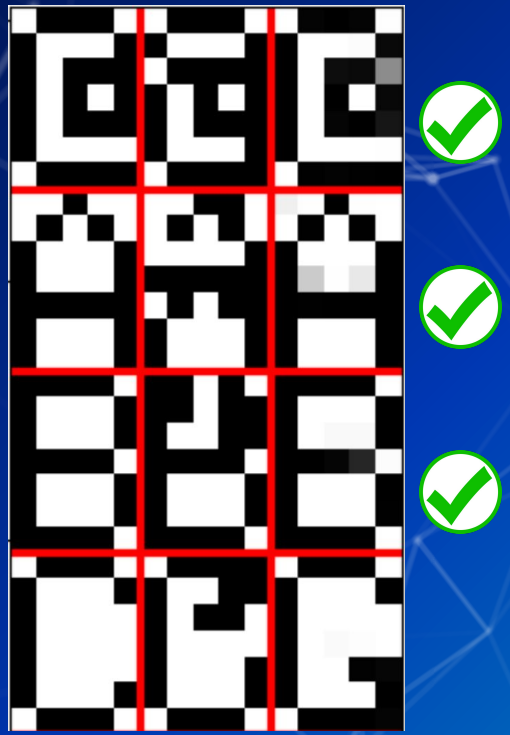
50% del conjunto de entrenamiento - Probabilidad de ruido: 10%

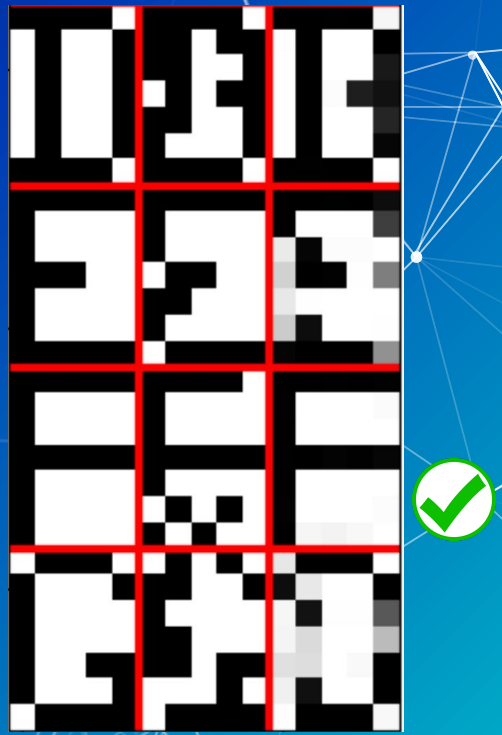




Denoising Autoencoder - Resultados

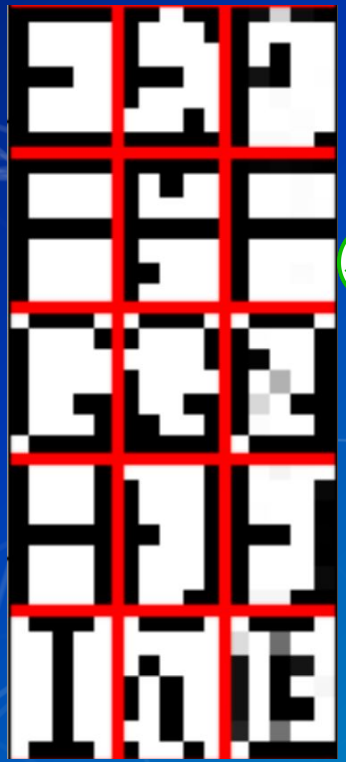
25% del conjunto de entrenamiento - Probabilidad de ruido: 15%

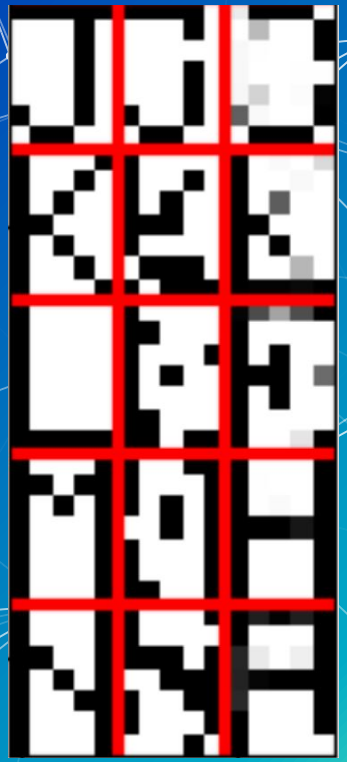




Denoising Autoencoder - Resultados

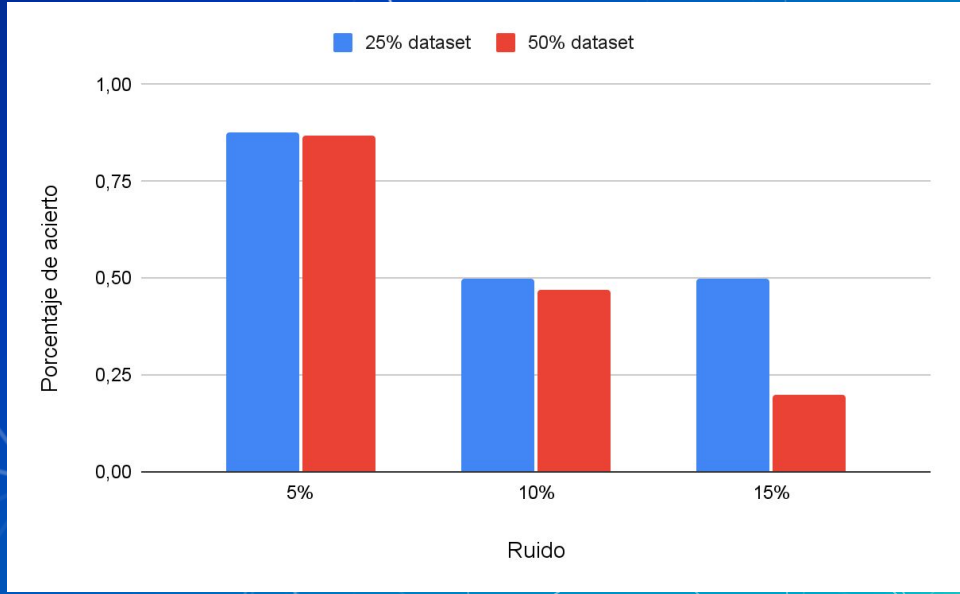
50% del conjunto de entrenamiento - Probabilidad de ruido: 15%





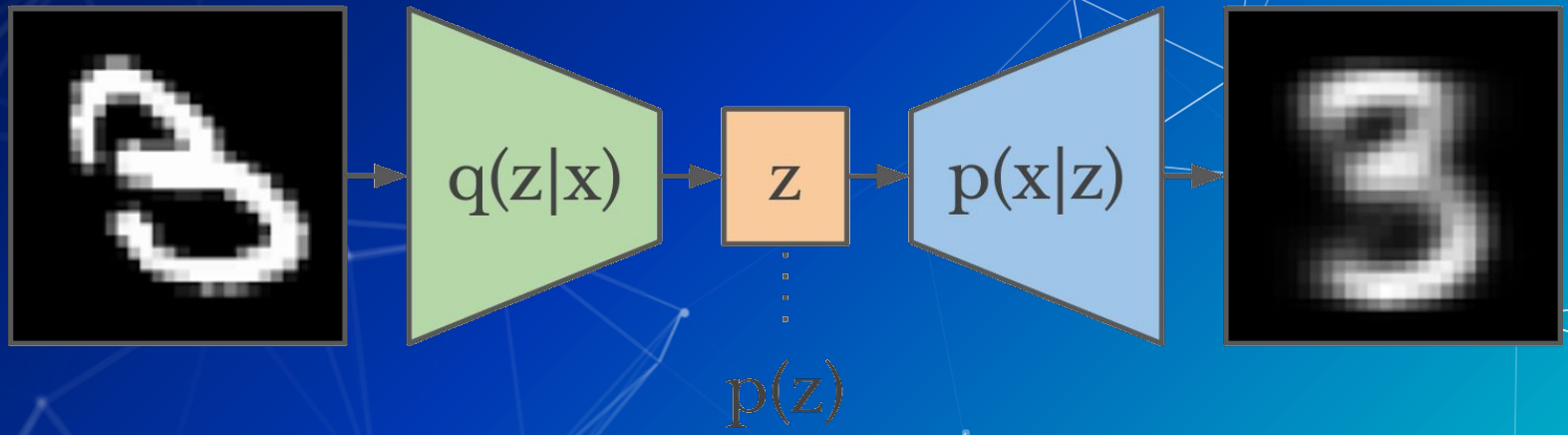
Denoising Autoencoder - Resultados

		Porcentaje del dataset entrenado	
		25%	50%
Ruido	5%	0,875	0,87
	10%	0,5	0,47
	15%	0,5	0,2



Autoencoder Variacional

Autoencoder variacional

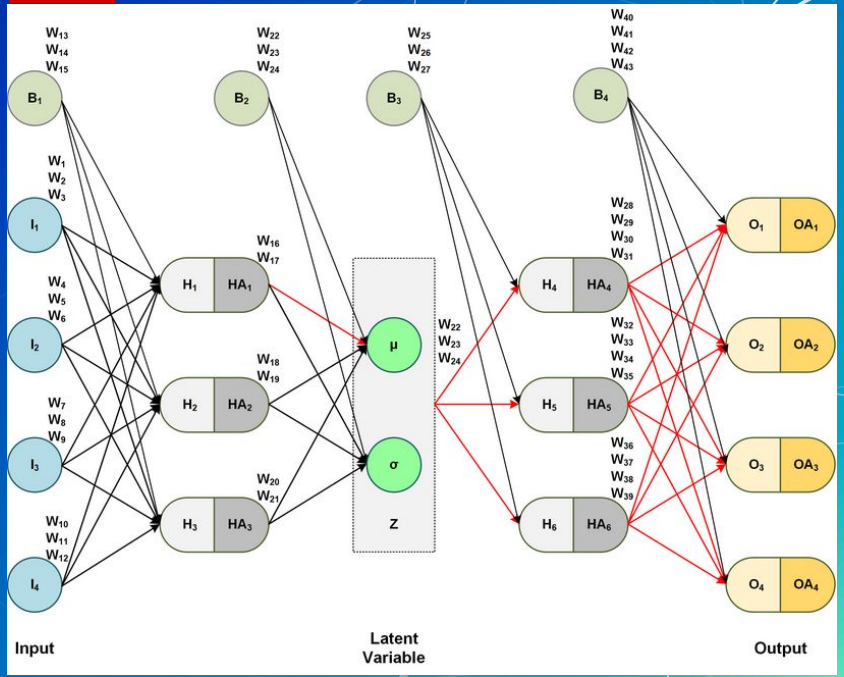


Vae 1: Letras de fonts

Estructura elegida

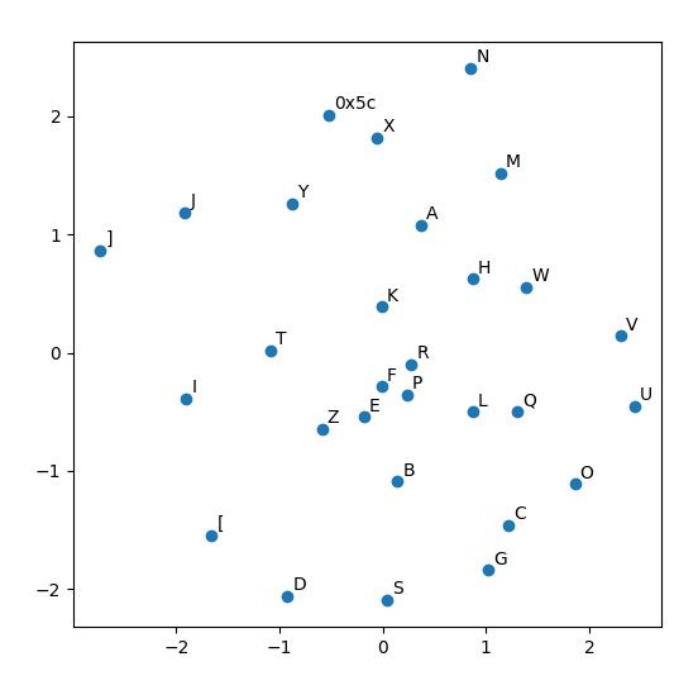
- Capa de entrada y salida: 35
- Encoder:
  - Capa intermedia: 16
  - Activación: Relu
- Capa latente(z): 2
- Sampling:
  - Media
  - Varianza logaritmica
- Decoder:
  - Capa intermedia 16
  - Activacion: Relu
- Capa de salida: sigmoide

Keras

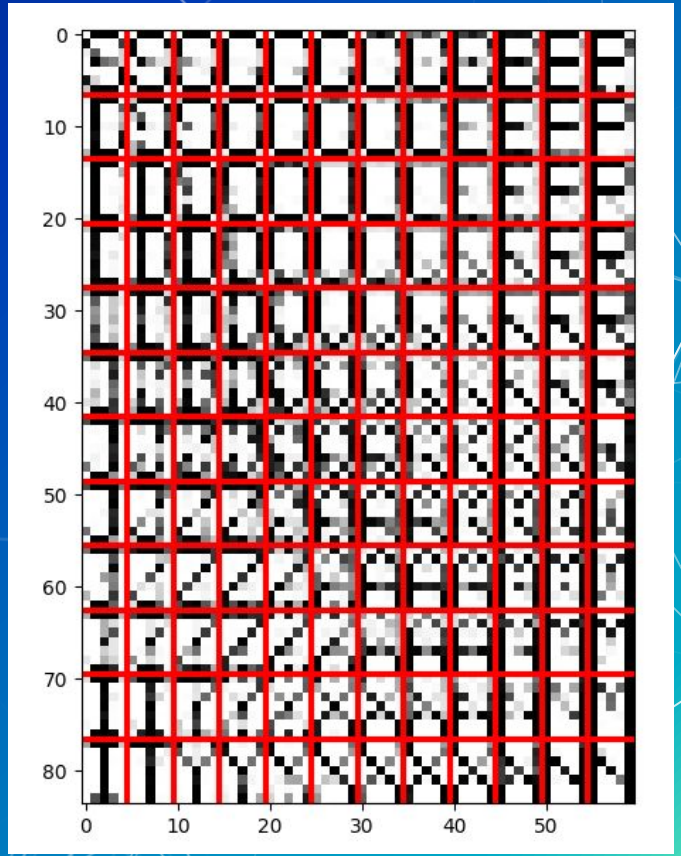


Vae1: Resultados

Espacio latente



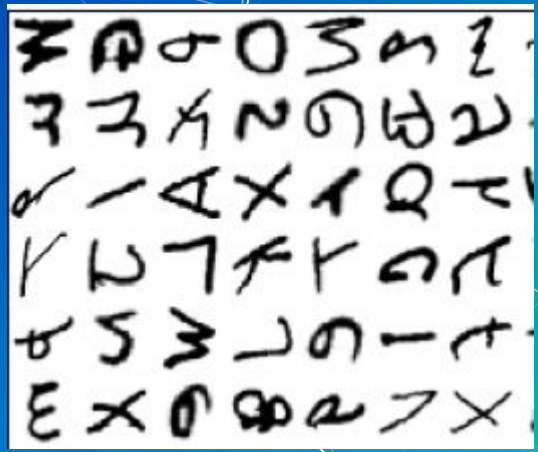
Generación de letras



Vae 2: Letras

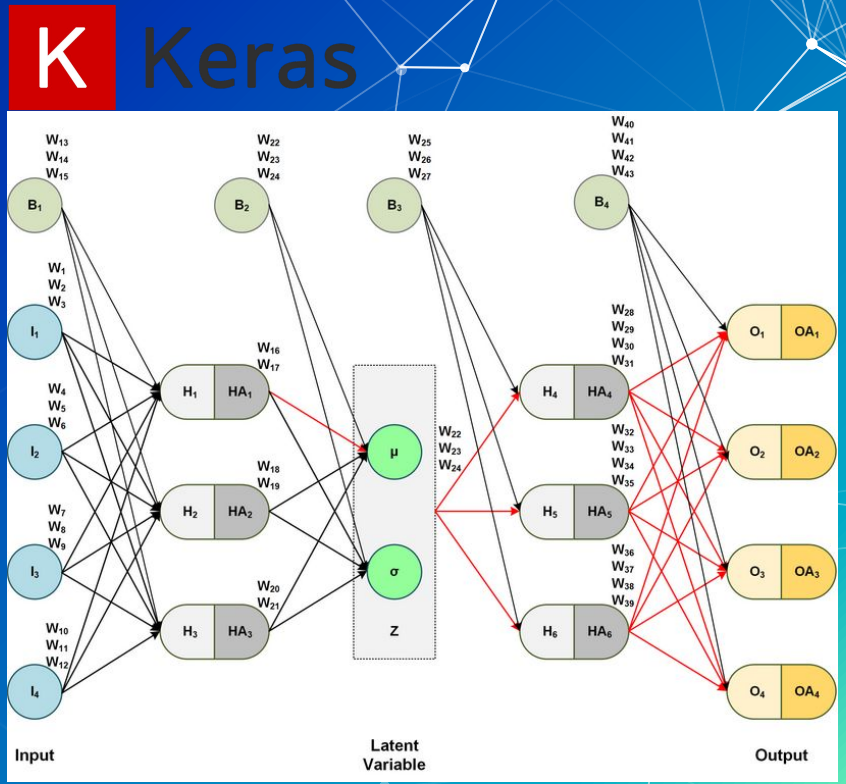
Vae 2: Datos de entrada

- Base datos de letras de EMINST
  - https://www.nist.gov/itl/products-and-services/emnist-dataset
- 145600 caracteres
- 26 clase
- Tamaño de (28, 28)
- 26 tipos de letras(MUY diferentes entre sí)



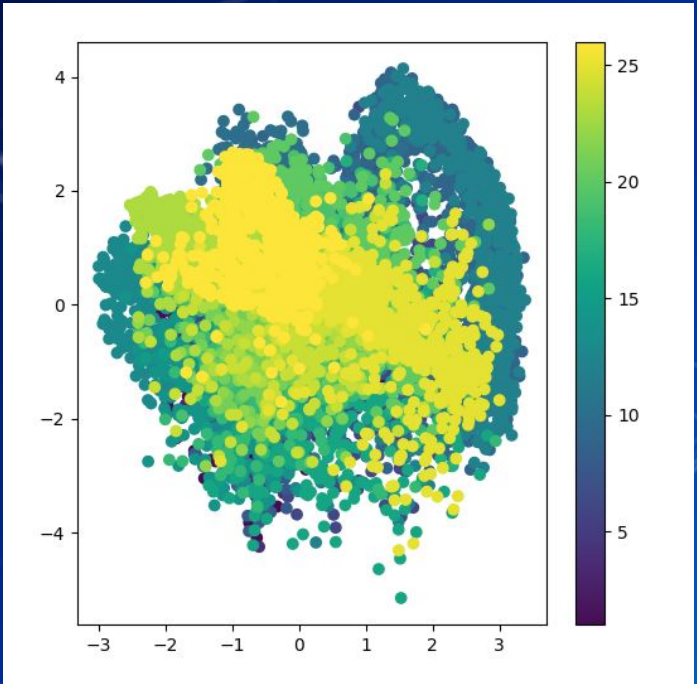
Vae 2: Estructura elegida

- Capa de entrada y salida: 28 * 28
- Encoder:
  - Capa intermedia: 256
  - Activación: Relu
- Capa latente(z): 2
- Sampling:
  - Media
  - Varianza logaritmica
- Decoder:
  - Capa intermedia 256
  - Activacion: Relu
- Capa de salida: sigmoide

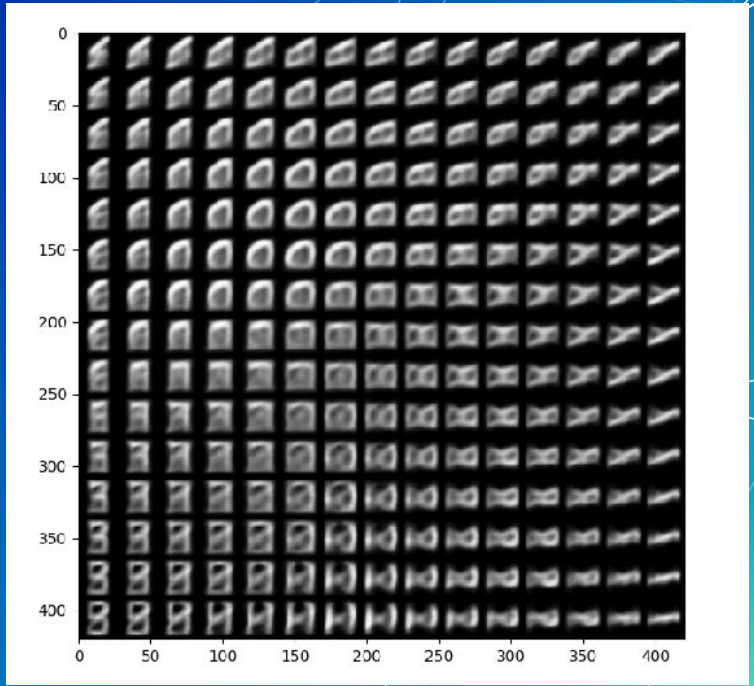


Vae 2: resultados

Espacio latente



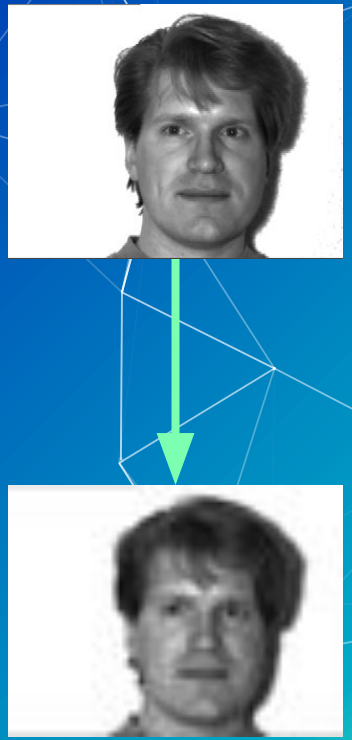
Generación de letras



Vae 3: 'Mapa de emociones'

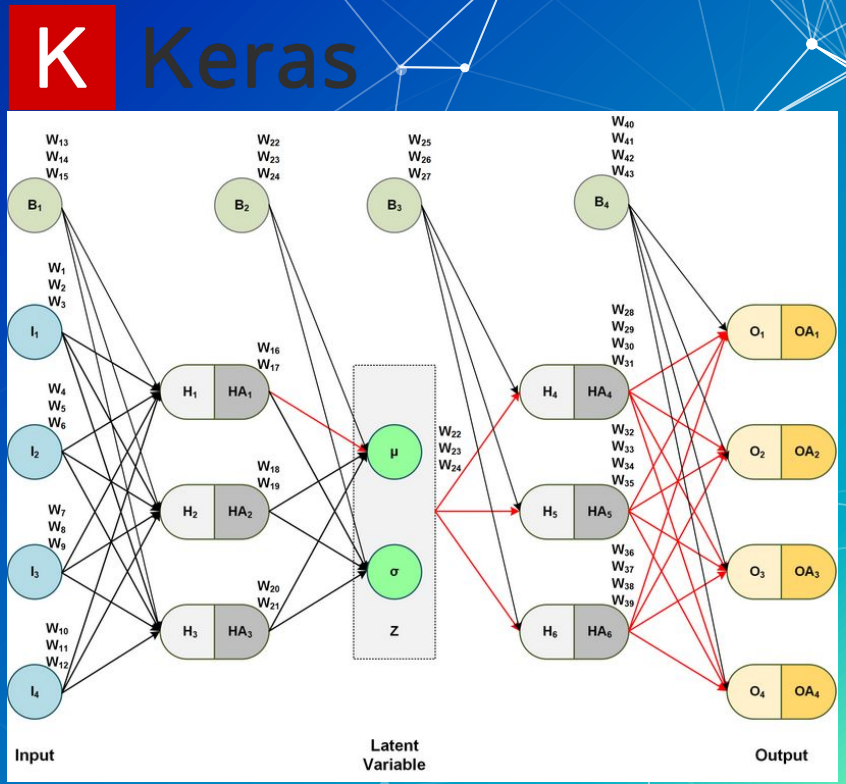
Vae 3: Datos de entrada

- Base de imágenes de caras de yalefaces
  - http://vision.ucsd.edu/content/yale-face-database
- Tamaño de original de (320, 243)
- Conversión a (50, 38)
- 5 emociones del mismo sujeto:
  - Neutro, feliz, triste, sorprendido, guiño, sueño.
- Queremos 'mapear' sus emociones moviéndonos en el espacio latente



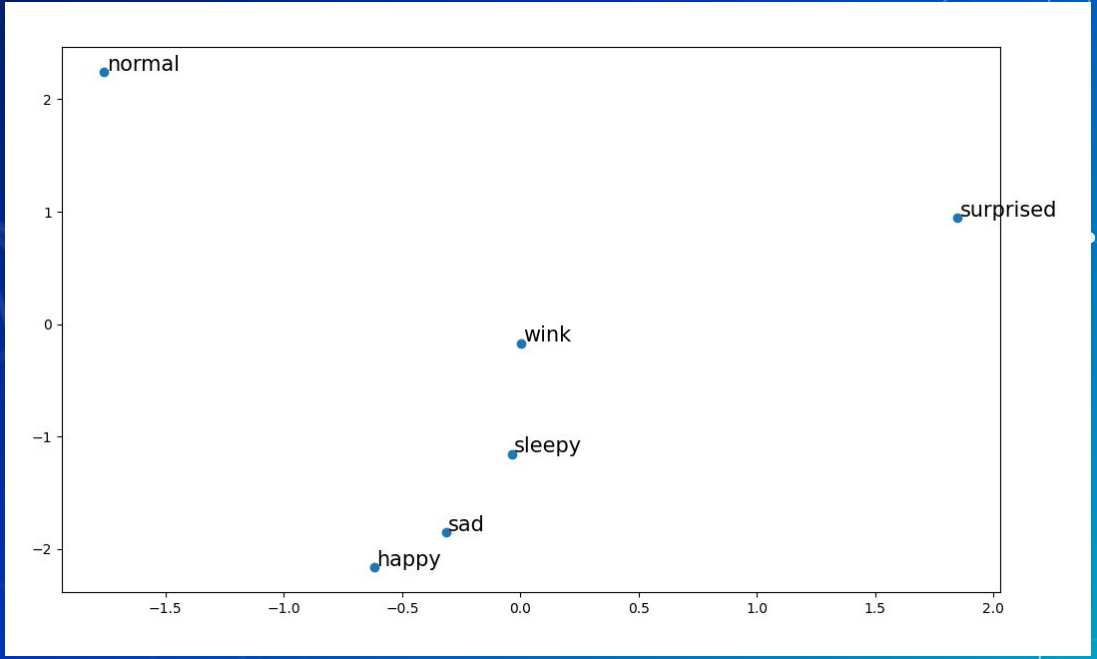
Vae 3: Estructura elegida

- Capa de entrada y salida: 64 * 50
- Encoder:
  - Capa intermedia: 1024
  - Activación: Relu
- Capa latente(z): 2
- Sampling:
  - Media
  - Varianza logaritmica
- Decoder:
  - Capa intermedia 1024
  - Activacion: Relu
- Capa de salida: sigmoide

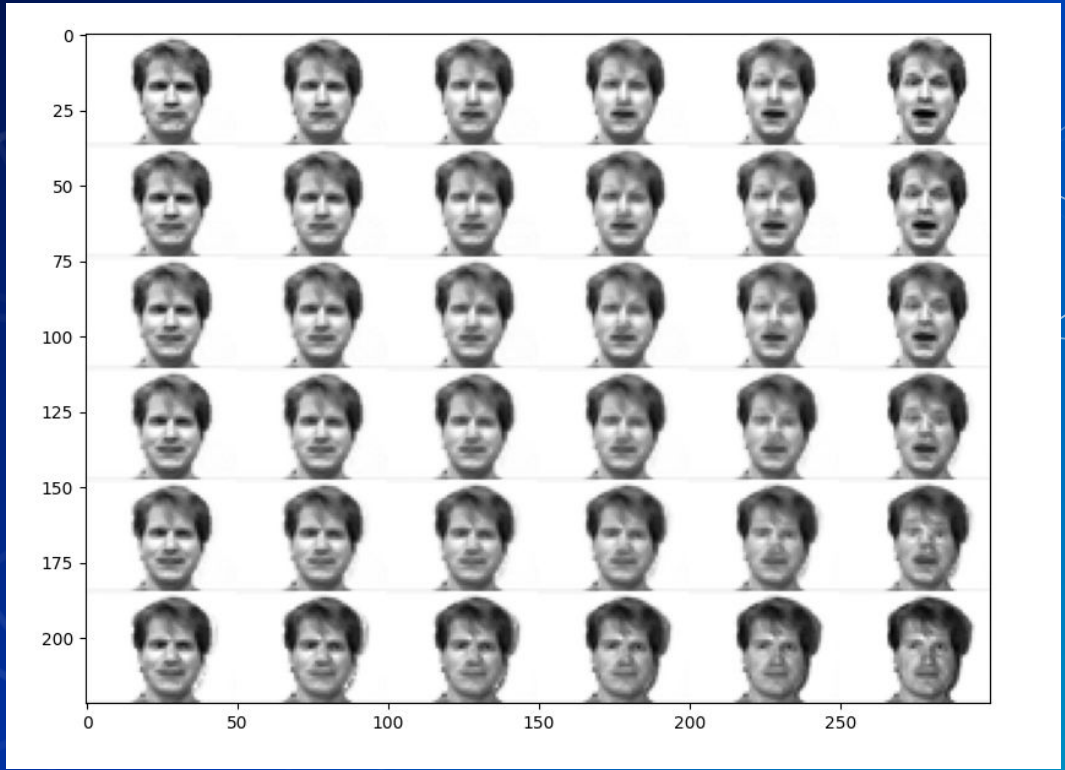


Vae 3: Resultados

Espacio latente

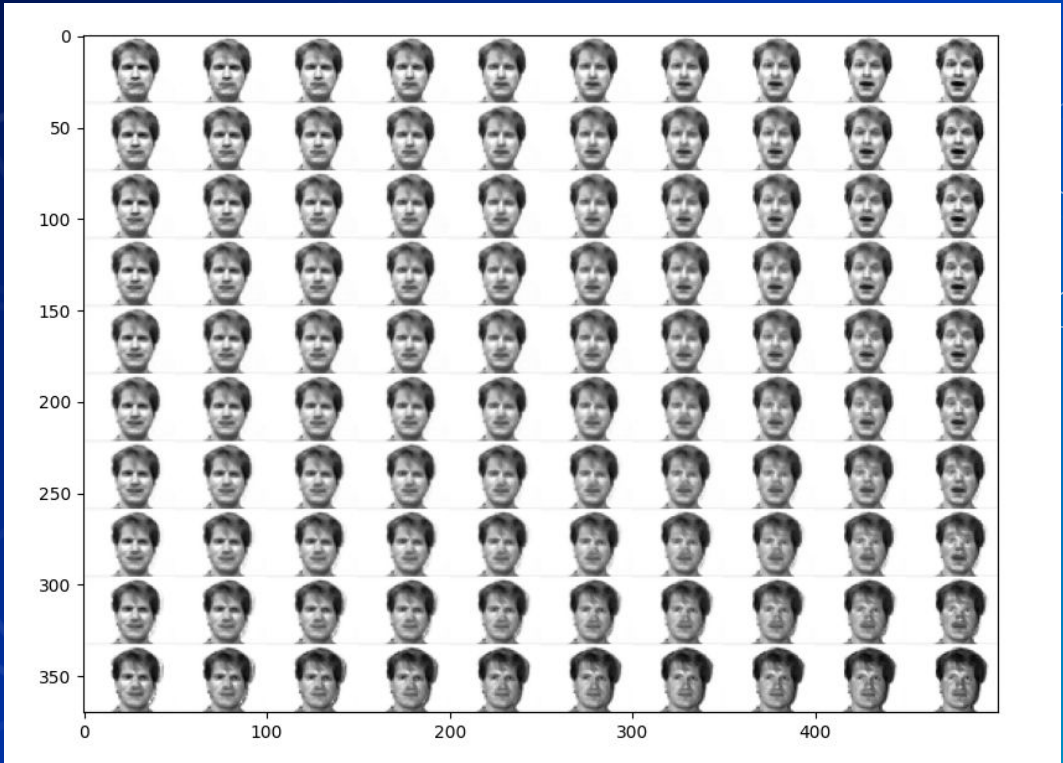


Vae 3: Resultados



Generación de 'emociones' moviéndonos en el espacio latente

Vae 3: Resultados



Generación de 'emociones' moviéndonos en el espacio latente

Conclusiones

Conclusiones particulares

Autoencoders lineales

- Los autoencoders lineales no son óptimos para la generación a través del espacio latente.
- Los autoencoders lineales son óptimos para disminuir la dimensión del input, siempre y cuando la configuración utilizada se adapte al conjunto a entrenar.

Denoising autoencoders

- Los DAE pueden ser muy efectivos para eliminar ruido del tipo salt & pepper

Autoencoder variacionales

- Los VAE son muy útiles para generar nuevos datos
- Son muy sensibles a la cantidad de datos y uniformidad del mismo

Conclusiones generales

- La cantidad de muestras que puede contener el conjunto de entrenamiento para un aprendizaje correcto depende estrictamente de la estructura del espacio latente.
- La pérdida de información de un autoencoder es inevitable y depende del conjunto de entrenamiento.
- La estructura del perceptrón multicapa es crucial para la minimización del error.
- Las optimizaciones permiten una minimización del error en menos iteraciones y también evitan el estancamiento en un mínimo local.

Mejoras posibles

1. Mayor optimización para el autoencoder lineal (utilizar Powell)
2. Conjunto de datos más robusto para el VAE de emociones(Misma cara más emociones)
3. Mejorar hardware (muchas simulaciones las tuvimos que cortar por tiempo)


¡Gracias!