# Informe: Redes Neuronales Convolucionales - UTN-FRD

Practicantes de **Escuela Educacion Tecnica Nº1 Luciano Reyes**.

**Alumnos**

1. Leiva Ivan.
2. Michel Leandro.

**Índice**

1. **Introducción**
2. **Descripción General del Proyecto**
3. **Introducción a las Redes Neuronales**
4. **Preparación del Entorno para la Toma de Fotos**
5. **Creación de Datasets**
6. **Primeros Pasos con Redes Neuronales**
7. **Introducción a Redes Neuronales Convolucionales (CNN)**
8. **Desarrollo y Entrenamiento de nuestra CNN**
9. **Conclusiones**
10. **Recomendaciones y Trabajo Futuro**
11. **Anexos**
12. **Referencias**

—————————————————————————————————————

## Introducción

1. **Objetivo de la práctica**

El objetivo principal de esta práctica profesionalizante fue utilizar la captura de imágenes mediante un celular para crear nuestros propios datasets y así entrenar una red neuronal convolucional (CNN) capaz de contar objetos. Esta experiencia buscó proporcionarnos una introducción práctica y teórica en el campo de las redes neuronales y el aprendizaje automático, áreas cruciales en el desarrollo actual de la inteligencia artificial.

1. **Importancia del proyecto en el contexto actual**

En un mundo cada vez más orientado hacia el uso de tecnologías inteligentes, la capacidad de diseñar y entrenar modelos de aprendizaje automático se ha convertido en una habilidad esencial. Las CNNs, en particular, son herramientas poderosas para el procesamiento y análisis de imágenes, con aplicaciones que van desde la medicina hasta la seguridad y la automatización industrial. Este proyecto nos permitió adentrarnos en estas tecnologías, desarrollando habilidades que son altamente valoradas en el mercado laboral actual.

1. **Agradecimientos**

Queremos expresar nuestro agradecimiento a Santiago Cerrotta por su guía y apoyo durante toda la práctica. Su experiencia y paciencia fueron fundamentales para nuestro aprendizaje. También agradecemos a la Universidad Tecnológica Nacional - Facultad Regional Delta (UTN-FRD) y al Grupo de Fotónica Aplicada por brindarnos esta oportunidad y los recursos necesarios para llevar a cabo el proyecto.

## 2. Descripción General del Proyecto

1. **Metas y objetivos**

La meta principal del proyecto era desarrollar un modelo de CNN que pudiera contar objetos en imágenes capturadas con un celular. Para ello, nos propusimos:

- Aprender los conceptos básicos de las redes neuronales.

- Diseñar una metodología para la captura de imágenes.

- Crear datasets variados con fotos de maíces, lentejas y arroz.

- Entrenar y evaluar diferentes arquitecturas de redes neuronales convolucionales.

- Alcanzar una precisión mínima del 90% en la detección y conteo de objetos.

1. **Descripción del entorno y equipos utilizados**

El proyecto se llevó a cabo en el laboratorio del grupo de fotónica aplicada de la UTN-FRD. Para la captura de imágenes, utilizamos un celular Samsung A21s con una resolución de 2250x4000 píxeles. Construimos un trípode con madera para asegurar la consistencia en la altura y el ángulo de las fotos, posicionando el celular a 16 cm sobre un fondo de cartulina negra. Las condiciones de iluminación se ajustaron apagando el foco superior y encendiendo los focos frontales para obtener una iluminación uniforme y minimizar sombras.



1. **Proceso de trabajo y metodología**

El trabajo se estructuró en varias fases:

1. **Introducción teórica:** Recibimos una charla introductoria sobre redes neuronales, cubriendo conceptos fundamentales como neuronas, pesos, capas y funciones de activación.
2. **Preparación del dispositivo de captura de fotos:** Construimos el trípode y ajustamos las condiciones de iluminación y posición del celular.
3. **Captura de imágenes:** Tomamos múltiples fotos de diferentes combinaciones de maíces, lentejas y arroz, moviendo los objetos asegurando la variabilidad de la información contenida en cada foto.
4. **Creación del dataset:** Procesamos las imágenes, reduciendo su resolución para optimizar el almacenamiento y manejo de datos.
5. **Entrenamiento de modelos:** Utilizamos librerías como Tensor Flow y Keras para diseñar y entrenar varias arquitecturas de CNN, evaluando su rendimiento con nuestros datasets personalizados.
6. **Evaluación y mejora:** Iteramos sobre diferentes modelos y parámetros hasta alcanzar una precisión satisfactoria en el conteo de objetos.

## 3. Introducción a las Redes Neuronales

1. **Conceptos básicos**

Al inicio de la práctica, recibimos una introducción a los conceptos fundamentales de las redes neuronales, una tecnología clave en el campo de la inteligencia artificial. Aprendimos que una red neuronal está compuesta por neuronas, las unidades básicas de procesamiento, que están conectadas por pesos. Estas neuronas se organizan en capas: una capa de entrada, una o más capas ocultas y una capa de salida.

Cada neurona recibe una entrada, la procesa mediante una función de activación y genera una salida. Los pesos determinan la influencia de cada entrada en la salida de la neurona. Además, las redes pueden incluir un término de bias que ajusta la salida de la neurona independientemente de las entradas.

También aprendimos sobre hiper parámetros importantes como los epochs, que definen el número de veces que el algoritmo de entrenamiento procesa todo el dataset, y el batch size, que determina el número de muestras que se procesan antes de actualizar los pesos de la red.

1. **Primera toma de contacto y aprendizaje inicial**

Nuestra primera experiencia práctica con redes neuronales consistió en entrenar un modelo simple para predecir la probabilidad de que una persona contraiga diabetes en los próximos cinco años. Utilizamos un dataset ya

preexistente, donde cada línea contiene ocho datos separados por comas: siete valores numéricos de análisis de diabetes y un valor binario indicando si la persona tiene diabetes (1) o no (0).

Programamos y entrenamos el modelo, logrando una precisión del 80%. Este ejercicio nos proporcionó una comprensión práctica de cómo funcionan las redes neuronales y cómo se pueden aplicar a problemas de clasificación.

Así mismo también entendimos conceptos como el overfitting que vendría a ser el sobredimensionar las cosas y que esto nos de resultados opuestos a los resultados buscados ya que a veces una arquitectura compleja no es necesaria para obtener resultados positivos.

## 4. Preparación del Entorno para la Toma de Fotos

1. **Construcción del trípode y preparación del entorno**

Para asegurar la consistencia en la toma de fotos, construimos un trípode artesanal utilizando madera. Este trípode fue diseñado para mantener el celular a una altura fija de 16 cm sobre un fondo de cartulina negra. La estabilidad y la consistencia en la posición del celular eran cruciales para obtener imágenes con características similares uniformes para el armado

apropiado del dataset.

1. **Condiciones de la toma de fotos**

Las fotos se tomaron en el laboratorio del Grupo de Fotónica Aplicada, con el celular en posición boca abajo y la cámara orientada hacia las tres mesas delanteras. Las condiciones de iluminación fueron cuidadosamente controladas: el foco directamente sobre nosotros estaba apagado, mientras que los demás focos del laboratorio estaban encendidos para proporcionar una iluminación uniforme.

Para asegurar que cada foto incluyera un marco consistente, dejamos visible entre 1 y 2 cm del borde del trípode en cada imagen. Este marco sirvió como referencia visual para mantener la uniformidad en la toma de foto y evitar

capturar un objeto incompleto.

1. **Configuración del celular y reducción de resolución**

Utilizamos un celular Samsung A21s con una resolución de 2250 x 4000 píxeles para capturar las imágenes. Para optimizar el procesamiento y almacenamiento de las imágenes, reducimos su resolución en un factor de 10 utilizando scripts en Python, obteniendo imágenes de 225 x 400 píxeles y aproximadamente 15 KB por foto.

Este tamaño reducido de las imágenes facilitó el manejo del dataset durante el entrenamiento de la red neuronal. Capturamos 50 fotos por cada cantidad de maíces, desde 1 hasta 50, moviendo los maíces entre cada toma para asegurar la variabilidad en el dataset.

## 

## 5. Creación de Datasets

1. **Toma de fotos de maíces**

Inicialmente, nos concentramos en capturar imágenes de diferentes cantidades de maíces. Utilizando el trípode y las condiciones de iluminación establecidas, comenzamos a tomar fotos con un celular Samsung A21s. Para cada cantidad de maíces, desde 1 hasta 50, tomamos 50 fotos, moviendo los maíces de forma aleatoria entre cada toma para asegurar la variabilidad. Las imágenes capturadas se redujeron a una resolución de 225 x 400 píxeles, resultando en archivos de aproximadamente 15 KB cada uno.

Al final se usaron únicamente las fotos de 1 a 20 maíces, 50 fotos por cantidad de maíz quedando un total de 1000 fotos.

1. **Toma de fotos de maíces y lentejas**

Después de completar las fotos de los maíces, incorporamos lentejas al proceso. Tomamos 10 fotos para cada combinación de maíces y lentejas, empezando con 1 maíz y de 1 a 10 lentejas, y así sucesivamente hasta llegar a 20 maíces y 10 lentejas, con un total de 2000 fotos. La inclusión de lentejas añadió un nivel adicional de complejidad y diversidad al dataset, asegurando que el modelo pudiera generalizar bien en escenarios con múltiples tipos de objetos.

1. **Toma de fotos de maíces, lentejas y arroz**

Finalmente, incluimos arroz en las fotos. Capturamos imágenes con combinaciones de maíces, lentejas y arroz, agregando un puñado de arroz al fondo. Este proceso también generó 2000 fotos, resultando en tres datasets principales: maíces solos (2500 fotos), maíces con lentejas (2000 fotos), y maíces con lentejas y arroz (2000 fotos). Cada foto se tomó con el mismo cuidado en cuanto a la posición y las condiciones de iluminación.

## 6. Introducción a Redes Neuronales Convolucionales (CNN)

1. **Conceptos básicos de CNN**

Luego de construir y organizar nuestros datasets, nos adentramos en el mundo de las redes neuronales convolucionales (CNN), que son especialmente efectivas para el procesamiento de imágenes. Aprendimos que una CNN está compuesta por capas de convolución, que aplican filtros (kernels) a las imágenes para extraer características relevantes. Estos filtros se entrenan para detectar patrones específicos, como bordes y texturas.

Las capas de maxpooling se utilizan para reducir la dimensionalidad de las representaciones intermedias, conservando las características más importantes mientras disminuyen el número de parámetros y el costo computacional. Capas adicionales como dropout ayudan a prevenir el sobreajuste durante el entrenamiento, y las capas flatten transforman las matrices de características en vectores que se pueden conectar a las capas densamente conectadas (fully connected).

1. **Prácticas con el dataset CIFAR10**

Para familiarizarnos con la implementación de CNNs, comenzamos trabajando con el dataset CIFAR 10, que contiene imágenes de 10 clases diferentes. Estudiamos cómo CIFAR10 organiza sus datos y preprocesar las imágenes, almacenadas en archivos .npy que contienen tanto las imágenes como sus etiquetas. Cada imagen se representa como un vector de 3072 posiciones (32x32 píxeles en 3 canales de color RGB).

Implementamos y entrenamos varias arquitecturas de CNN usando Tensor Flow y Keras, ajustando hiper parámetros como la cantidad de filtros, el tamaño de los kernels, y la tasa de dropout. Practicamos dividiendo el dataset en conjuntos de entrenamiento y prueba, y evaluando la precisión del modelo.

Finalmente, utilizamos el conocimiento adquirido para crear y entrenar nuestras propias CNNs con los datasets de maíces, lentejas y arroz. Tras múltiples pruebas y ajustes, logramos alcanzar una precisión del 96% en el conteo de objetos, cumpliendo así los objetivos de la práctica profesionalizante.

## 7. Desarrollo y Entrenamiento de nuestra CNN

1. **Creación del dataset con nuestras imágenes**

Con las imágenes capturadas, organizamos nuestro dataset en tres categorías: maíces solos, maíces con lentejas y maíces con lentejas y arroz. Cada categoría se almacenó en carpetas separadas, etiquetadas adecuadamente para facilitar el proceso de entrenamiento. Este dataset personalizado nos permitió entrenar la red neuronal con datos específicos y relevantes para nuestro objetivo.

1. **Diseño y pruebas de arquitecturas de red**

Entrenamos varias arquitecturas de redes neuronales convolucionales (CNN) utilizando Tensor Flow y Keras. Se partió el dataset en 80% entrenamiento y 20% testeo y probamos diferentes configuraciones de capas de convolución, maxpooling y dropout, buscando la combinación óptima que nos proporcionará la mejor precisión. Ajustamos hiper parámetros como el tamaño de los filtros variando de 8 hasta 128, el número de capas añadiendo hasta 4 capas convolucionales, strides y la tasa de aprendizaje.

**La mejor arquitectura lograda fue:**

Conv2D(32, (3, 3), padding='same', activation='relu', MaxPooling2D(pool\_size=(2, 2)))

(Dropout(0.25))

(Conv2D(64, (3, 3), padding='same', activation='relu'))

(MaxPooling2D(pool\_size=(2, 2)))

(Dropout(0.25))

(Flatten())

(Dense(1024, activation='relu'))

(Dropout(0.15))

(Dense(num\_classes, activation='softmax'))

1. **Resultados obtenidos**

Después de numerosos ensayos y ajustes, logramos alcanzar una precisión del 96% en el conteo de objetos. El entrenamiento se realizó en sesiones de 20 minutos, con 10 epochs y un batch size de 35. Este nivel de precisión superó nuestro objetivo inicial del 90%, demostrando la efectividad de nuestras arquitecturas y métodos de entrenamiento.

| conv2d, 32f, 3x3, relu  maxPooling 2x2  Dropout 0.25  conv2d 64f, 3x3, relu  maxPooling 2x2  Dropout 0.25  Flatten  Dense 512n, relu  (Usando doble Dense 512n baja a 92,50%)  Dropout 0.15  Dense (numClases)n softmax | Epochs: 5  BatchSize: 32  Time x Epoch: 53s  Total time: 4.41min  num Imagenes: 1000 RGB  resolución: 225alto x 400ancho  num Clases: 20 (50 fts x 20 class)  testAccuracy: 94,00% |
| --- | --- |

| conv2d, 16f, 3x3, relu  maxPooling 2x2  Dropout 0.25  conv2d 32f, 3x3, relu  maxPooling 2x2  Dropout 0.25  Flatten  Dense 512n, relu  Dropout 0.15  Dense (numClases)n softmax | Epochs: 5  BatchSize: 32  Time x Epoch: 22s  Total time: 1.83min  num Imagenes: 1000 RGB  resolución: 225alto x 400ancho  num Clases: 20 (50 fts x 20 class)  testAccuracy: 93,50% |
| --- | --- |

| conv2d, 32f, 3x3, relu  maxPooling 2x2  Dropout 0.25  conv2d 64f, 3x3, relu  maxPooling 2x2  Dropout 0.25  Flatten  Dense 1024n, relu  Dense 512n, relu  Dropout 0.15  Dense (numClases)n softmax | Epochs: 5  BatchSize: 32  Time x Epoch: 83s  Total time: 6.9min  num Imagenes: 1000 RGB  resolución: 225alto x 400ancho  num Clases: 20 (50 fts x 20 class)  testAccuracy: 91,50% |
| --- | --- |

| conv2d, 32f, 3x3, relu  maxPooling 2x2  Dropout 0.25  conv2d 64f, 3x3, relu  maxPooling 2x2  Dropout 0.25  Flatten  Dense 1024n, relu  (Usando Dense 2048n Crasheo)  Dropout 0.15  Dense (numClases)n softmax | Epochs: 5  BatchSize: 32  Time x Epoch: 78s  Total time: 6.5min  num Imagenes: 1000 RGB  resolución: 225alto x 400ancho  num Clases: 20 (50 fts x 20 class)  testAccuracy: 95,00% |
| --- | --- |

## 

## 8. Empezando una nueva tanda de fotos

1. **Inclusión de arroz en las imágenes**

Luego de finalizar las 2000 fotos de maíces y lentejas, decidimos añadir arroz

al proceso. Capturamos otras 2000 fotos, incluyendo combinaciones de

maíces, lentejas y arroz. Las imágenes adicionales enriquecieron nuestro

dataset, permitiéndonos entrenar y probar la robustez de nuestra CNN en

escenarios más complejos.

1. **Resultados de la nueva tanda de fotos**

El nuevo dataset, compuesto por 2000 fotos adicionales, se sumó a los

datasets previos, completando un total de 6500 imágenes. Estas imágenes

nos permitieron realizar pruebas adicionales y afinar aún más nuestras

arquitecturas de CNN. La inclusión de arroz añadió una capa de complejidad

que desafió a nuestros modelos, mejorando su capacidad para generalizar y

manejar variabilidad en los datos.

1. **Conclusión de los datasets**

Concluimos con tres datasets bien estructurados: uno de maíces solos (2500

fotos), uno de maíces con lentejas (2000 fotos) y uno de maíces con lentejas

y arroz (2000 fotos). Estos datasets nos proporcionaron una base sólida para

entrenar y evaluar nuestras redes neuronales, demostrando la importancia de

un dataset diversificado y bien organizado.

## 9. Conclusiones

1. **Evaluación del trabajo realizado**

La práctica profesionalizante fue un éxito, permitiéndonos adquirir conocimientos sólidos sobre redes neuronales y su aplicación en el reconocimiento de objetos. El proceso de captura de imágenes, creación de datasets y entrenamiento de CNNs nos brindó una experiencia práctica invaluable.

1. **Conocimientos adquiridos**

Aprendimos a construir y entrenar redes neuronales convolucionales desde cero, comprender su funcionamiento interno y ajustar sus parámetros para mejorar la precisión. También adquirimos habilidades en la manipulación de datos, preprocesamiento de imágenes y evaluación de modelos.

1. **Impacto del proyecto en nuestra formación**

Esta práctica nos proporcionó una introducción completa al campo de la inteligencia artificial y el aprendizaje automático. Aunque nuestro conocimiento es básico, sentamos una base sólida para futuras investigaciones y aplicaciones en el campo de las redes neuronales y la inteligencia artificial. La experiencia adquirida es un primer paso importante en nuestra formación profesional, abriendo puertas a nuevas oportunidades y desafíos en el ámbito tecnológico.

## 10. Recomendaciones y Trabajo Futuro

1. **Sugerencias para futuras prácticas**

**Planificación Detallada:** Definir claramente los objetivos, pasos y cronograma del proyecto desde el inicio para asegurar un desarrollo fluido y eficiente.

**Mejorar la Calidad del Dataset:** Utilizar cámaras de mayor resolución y condiciones de iluminación controladas para mejorar la calidad de las imágenes.

**Ampliar la Variedad de Objetos:** Incluir una mayor diversidad de objetos para crear un dataset más robusto y desafiante.

**Automatización del Proceso:** Implementar herramientas automáticas para la toma de fotos y el etiquetado de imágenes, reduciendo el esfuerzo manual y mejorando la consistencia.

1. **Posibles mejoras en el proyecto**

**Ajuste de Hiper Parámetros:** Realizar una búsqueda más exhaustiva de hiper parámetros utilizando técnicas avanzadas como la búsqueda en cuadrícula o la optimización bayesiana.

**Aumento de Datos:** Aplicar técnicas de aumento de datos (data augmentation) para incrementar la variabilidad del dataset sin necesidad de capturar nuevas imágenes.

**Validación Cruzada:** Implementar validación cruzada para evaluar mejor la generalización del modelo y prevenir el sobreajuste.

**Transfer Learning:** Experimentar con transfer learning, utilizando modelos pre entrenados en datasets más grandes para mejorar la precisión y reducir el tiempo de entrenamiento.

## 11. Anexos

1. **Código fuente**

En esta sección se incluye el código fuente desarrollado durante la práctica, organizado por módulos y acompañado de comentarios detallados que explican su funcionamiento. El código se presenta en formato de notebooks de Jupyter, facilitando la replicación y el análisis de los resultados.

Descargue desde aquí el código fuente (GitHub)

1. **Ejemplos de imágenes tomadas**

Maíces solos (de 1 a 50 maíces).

Combinaciones de maíces y lentejas.

Combinaciones de maíces, lentejas y arroz.

Estas imágenes ilustran la variabilidad y la diversidad del dataset creado.

1. **Documentación adicional**

Instalación de librerías mediante CMD con ‘pip install {lib\_name}’. Ej: pip install tensorflow.

## 12. Referencias

1. **Bibliografía**

* [Playlist YT](https://www.youtube.com/playlist?list=PL-Ogd76BhmcC_E2RjgIIJZd1DQdYHcVf0)
* [Blog red neuronal para detectar diabetes](https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/)
* [Introducción redes convolucionales](https://bootcampai.medium.com/redes-neuronales-convolucionales-5e0ce960caf8)
* [GitHub sobre CNN de LA UTN-GFA](https://github.com/UTN-GFA/UTN-GFA.github.io)
* [GitHub sobre CIFAR10](https://gist.github.com/eblancoh/d379d92a3680360857581d8937ef114b)
* [Como entrenar una red con CIFAR 10](https://datasmarts.net/es/como-entrenar-una-red-neuronal-en-cifar-10-con-keras/)
* [Blog de funcionamiento de CIFAR10/100](https://www.cs.toronto.edu/%7Ekriz/cifar.html)
* [Como crear un dataset similar a CIFAR10](https://stackoverflow.com/questions/35032675/how-to-create-dataset-similar-to-cifar-10)
* [Keras web oficial](https://www.tensorflow.org/guide/keras?hl=es-419)