UNIVERSITAT POLITÈCNICA DE VALÈNCIA

CVBLAB

DEPARTAMENTO DE COMUNICACIONES

CENTRO FORMACIÓN PERMANENTE

Deep Learning aplicado al análisis de señales e imágenes

μPROYECTOS

Contenidos

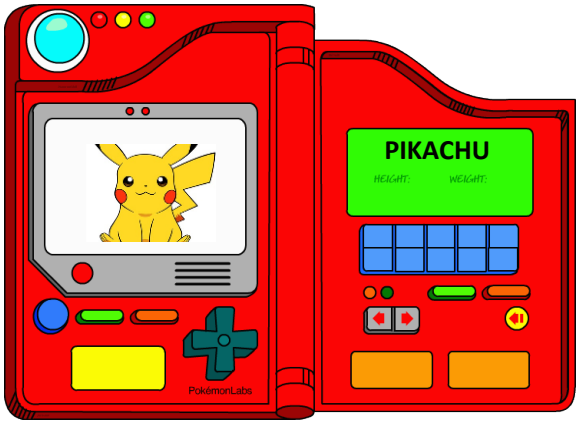
1. Nivel básico: Implementación de una Pokedex
2. Nivel intermedio: Desarrollo de un motor de búsqueda de ropa
3. Nivel avanzado: Predicción del valor de una vivienda

Contenidos

1. **Nivel básico: Implementación de una Pokedex**
2. Nivel intermedio: Desarrollo de un motor de búsqueda de ropa
3. Nivel avanzado: Predicción del valor de una vivienda

Descripción del µproyecto

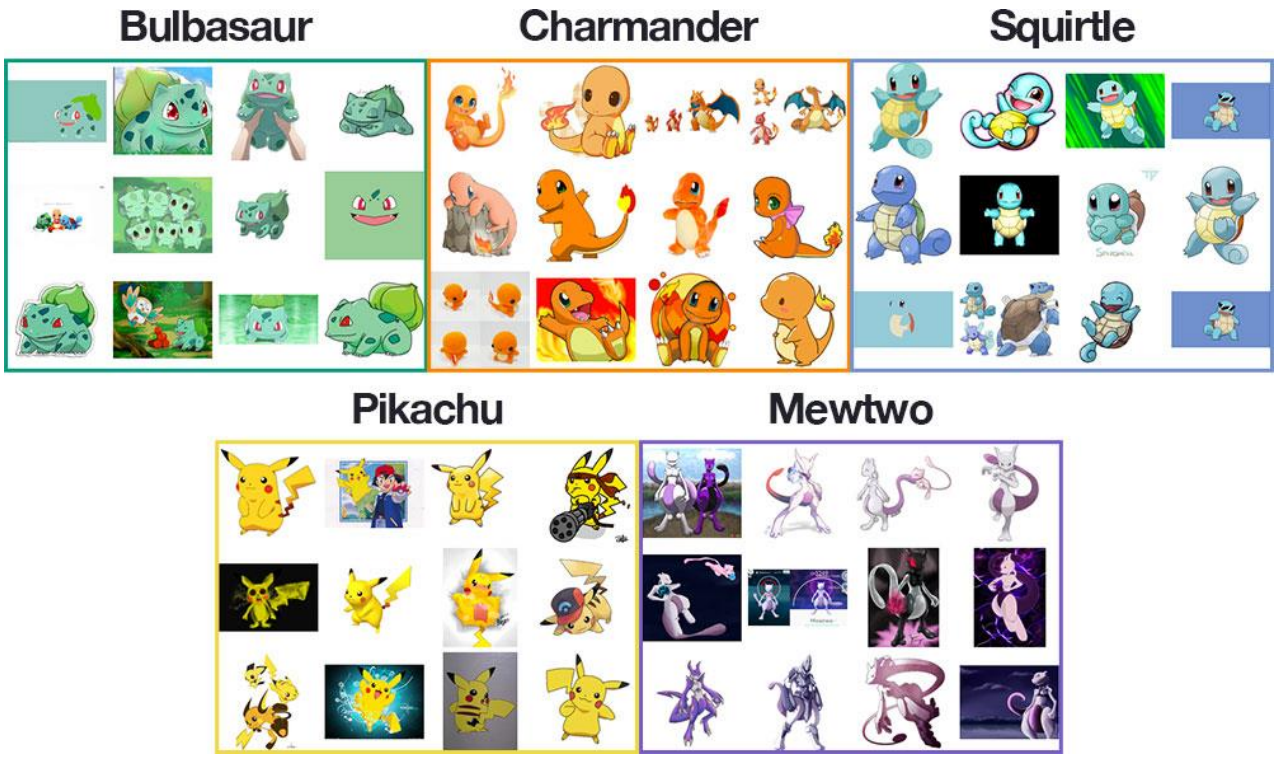
- Desarrollo de un **algoritmo de clasificación** basado en **redes neuronales convolucionales** capaz de discriminar entre **5 tipos diferentes de Pokemon**: Squirtle, Bulbasur, Charmander, Pikachu y Mewtwo.



- El modelo de predicción podrá ser **embebido** en una **aplicación** para **móvil** si se desea, emulando así el dispositivo Pokedex.

Descripción del µproyecto

- Disponemos de un **set de datos balanceado** que contiene **231** imágenes de **Squirtle**, **240** de **Bulbasur**, **245** de **Charmander**, **239** de **Pikachu** y **245** de **Mewtwo**.



Consejos para el desarrollo

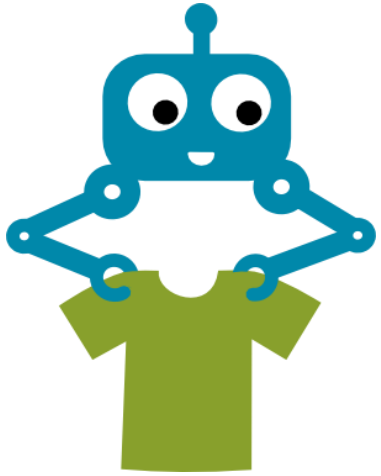
- Desarrollo del algoritmo empleando **Keras**.
- **Subir datos** a **Google Drive** y montar dicha unidad en **Colab para acceder** a los mismos.
- Crear **vector de etiquetas** (*y_label*) **a partir** del **nombre de los directorios** en los que están contenidos los datos.
- **Re-escalar** imágenes y **normalizarlas** (valores de pixels entre 0 y 1). **División** del conjunto de **datos** en training (80%) y validación (20%).
- Debido al escaso número de muestras de cada clase será necesario el uso de la técnica de generación sintética de datos: **Data augmentation**.
- Como **primera aproximación** se puede entrenar la **arquitectura de red convolucional** usada en la **práctica 4** para clasificar el dataset CIFAR10.
- A partir de los resultados obtenidos (*benchmark*) ir **aumentando la profundidad de la red** y quedarse con el modelo que presente los mejores resultados.
- **Re-entrenar arquitecturas de red ya existentes** en la literatura. **Comparar** los **resultados** obtenidos con los registrados en el punto anterior.
- **Evaluar el modelo** con **nuevas imágenes** que no se encuentren en el conjuntos de datos de entrenamiento.

Contenidos

1. Nivel básico: Implementación de una Pokedex
2. **Nivel intermedio: Desarrollo de un motor de búsqueda de ropa**
3. Nivel avanzado: Predicción del valor de una vivienda

Descripción del µproyecto

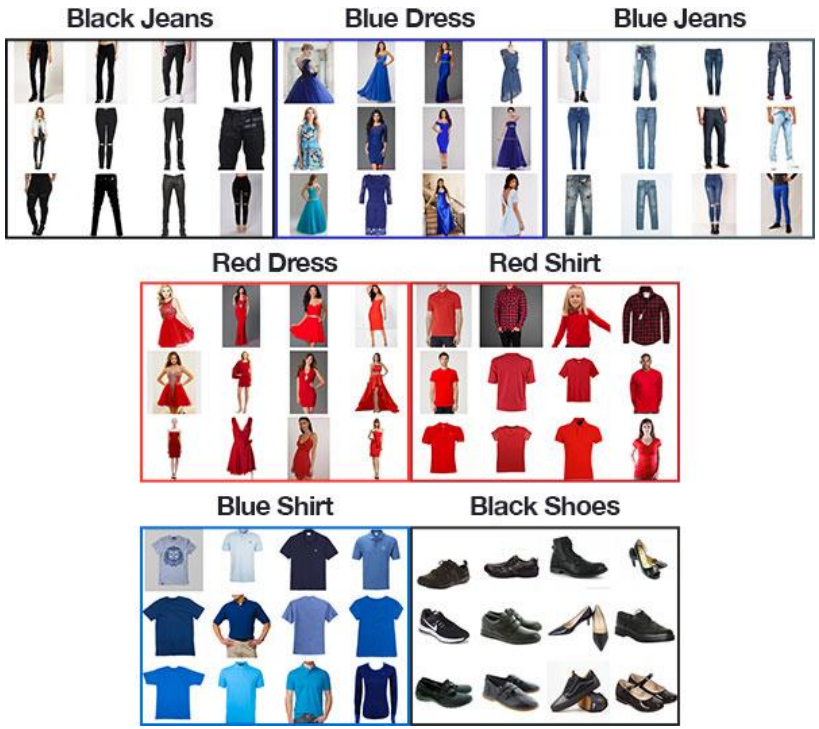
- Desarrollo de un **modelo de predicción capaz de estimar** el **tipo de ropa** y el **color** de la misma. Concretamente será capa de reconocer entre **vestidos, pantalones, camisetas** y **zapatos** de colores **negro, azul y rojo**.



- La **arquitectura de red convolucional** propuesta estará caracterizada por **dos ramas**, una referente al **tipo de prenda** y otro que será la encargada de clasificar el **color de la misma**.

Descripción del µproyecto

- Disponemos de un **set de datos balanceado** que contiene **334** ítems de **pantalones negros**, **391 zapatos negros**, **386 vestidos azules**, **356 pantalones azules**, **369 camisetas azules**, **380 vestidos rojos** y **332 camisetas rojas**.



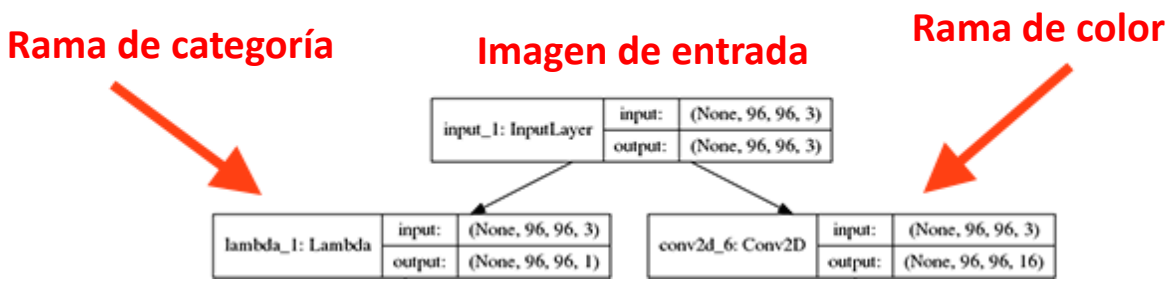
Consejos para el desarrollo

- Desarrollo del algoritmo empleando la **API funcional** de **Keras** ya que se trata de un **modelo con múltiples salidas**.
- **Subir datos** a **Google Drive** y montar dicha unidad en **Colab para acceder** a los mismos.
- Crear **dos vector de etiquetas** (*y_label_t* ; *y_label_c*) **a partir** del **nombre de los directorios** en los que están contenidos los datos.
- **Re-escalar** imágenes y **normalizarlas** (valores de pixels entre 0 y 1). **División** del conjunto de **datos** en training (80%) y validación (20%).
- Teniendo en cuenta el número de muestras probar el uso de la técnica de generación sintética de datos: ***Data augmentation***.
- **Evaluar el modelo** con **nuevas imágenes** que no se encuentren en el conjuntos de datos de entrenamiento. El modelo de predicción **debe** ser capaz de **estimar con éxito las 12 posibles combinaciones**.

Consejos para el desarrollo (arquitectura)

- Necesidad de proponer una **arquitectura** de **red neuronal convolucional con dos ramas**: **una** para clasificar el **tipo de ropa** y **otra** encargada de estimar el **color de una prenda**.
- La tarea de **detectar el color** es más sencilla que la de predecir el tipo de ropa, por lo que la **rama** correspondiente a la primera tarea será **menos profunda**.
- La **rama** encargada de predecir el **tipo de ropa** tan solo requiere como **entrada** la **imagen en escala de grises** mientras que **para estimar el color** será necesaria la **RGB**.
- Hacer uso de una **capa Lambda customizable** por nosotros mismos para **tomar** solo la información de **luminancia** en la rama encargada de predecir el tipo de prenda.

```
x = Lambda(lambda c: tf.image.rgb_to_grayscale(c))(inputs)
```



Contenidos

1. Nivel básico: Implementación de una Pokedex
2. Nivel intermedio: Desarrollo de un motor de búsqueda de ropa
3. **Nivel avanzado: Predicción del valor de una vivienda**

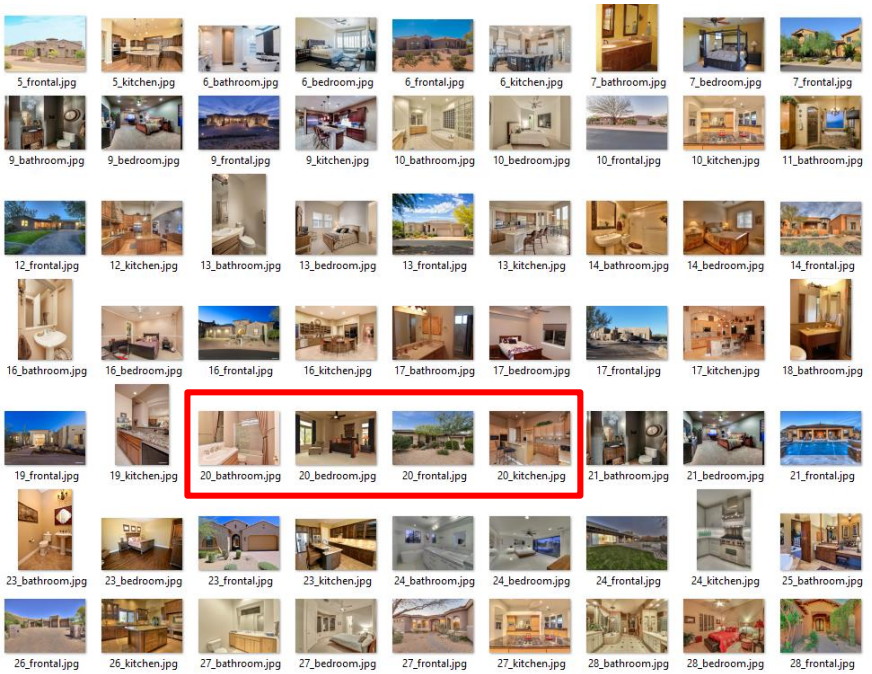
Descripción del µproyecto

- Desarrollo de un **modelo de predicción** del **valor** de un **inmueble** según ciertas **características numéricas**, **categóricas** y cuatro **imágenes** tomadas de la vivienda.

- La **arquitectura de red propuesta** tendrá que **combinar** la **información visual** de las imágenes con la **información numérica y categórica** de los datos. Para ello, se debe proponer una **red híbrida convolucional y perceptrón multicapa.**

Descripción del µproyecto

- Disponemos de un **set de datos de 535 casas** que **atendiendo** a su **código postal muestra signos de desbalanceo.** **Cada muestra** del conjunto de datos se caracteriza por **cuatro imágenes** (vista frontal exterior, baño, cocina y dormitorio), una serie de **datos o características** (i.e. #habitaciones, #baños, área, código postal) y nuestro ***target***, el precio.



	A	B	C	D	E
1	# Habitaciones	# Baños	Area	Código Postal	Precio
2	4	4	4053	85255	869500
3	4	3	3343	36372	865200
4	3	4	3923	85266	889000
5	5	5	4022	85262	910000
6	3	4	4116	85266	971226
7	4	5	4581	85266	1249000
8	3	4	2544	85262	799000
9	4	5	5524	85266	1698000
10	3	4	4229	85255	1749000
11	4	5	3550	85262	1500000
12	5	5	4829	85266	519200
13	4	4	3428	85255	1039000
14	5	3	5462	85266	799000
15	4	4	4021	85266	889000
16	5	5	4406	85266	700000
17	4	4	3721	85255	500000
18	5	3	3710	85331	740000
19	3	4	2748	85255	725000
20	5	4	4190	85255	1199000

Consejos para el desarrollo

- Desarrollo del algoritmo empleando la **API funcional** de **Keras** ya que se trata de un **modelo con múltiples entradas**.
- **Subir datos** a **Google Drive** y montar dicha unidad en **Colab para acceder** a los mismos.
- **Lectura** de archivo "***.csv***" que contiene los **datos numéricos y categóricos** mediante la librería **pandas**.
- ***Pruning*** de los **datos** según su código postal **para balancear** el dataset (i.e. casas pertenecientes a códigos postales conteniendo menos de 25 casas se eliminan del set de datos).
- **Escalar** las **características numéricas** (#habitaciones, #baños, área) al mismo rango de valores (i.e. entre 0 y 1) y **codificar** las **categóricas** (código postal) en ***one-hot-encoding***. **Concatenar** ambos tipos de características.
- Crear **el vector *target* (etiquetas de valor continuo)** que contendrá el **valor** de cada una de las **casas** (**escalado entre 0 y 1** dividiendo entre el máximo precio del set de datos)**.**

Consejos para el desarrollo

- **Lectura** de **imágenes**, **re-escalado**, creación del **montaje 2x2** y **normalización** (valores de pixels entre 0 y 1).
- ¿Por qué es **necesario realizar un montaje** en una sola imagen? ¿Qué problemas presenta el pasarle **imagen a imagen**?¿**Otras alternativas**?

Consejos para el desarrollo

- División del conjunto de **datos** en training (75%) y test (25%).
- No existe posibilidad de aplicar técnica de ***data augmentation*** debido a las características numéricas (existen técnicas de interpolación para generarlas pero no muy fiables) y categóricas (realmente delicadas).
- **Evaluar el modelo** con **la misma partición de test** debido a la **imposibilidad de recolectar nuevas muestras**. **No** existe la posibilidad de **subdividir el conjunto de entrenamiento** debido a la **escasez de datos**.

Consejos para el desarrollo (arquitectura)

- **Arquitectura de red** que **combine** la **información visual** de las imágenes con la **información numérica y categórica** de los datos. Para ello, se debe proponer una **red híbrida convolucional y perceptrón multicapa**.
- Será necesario crear la red empleando la **API funcional** puesto que tenemos dos tipos de entrada.
- Para poder concatenar ambas entradas, la **última capa densa de ambas ramas** tiene que tener el **mismo número de neuronas**.
- Como estamos haciendo frente a un **problema de regresión**, la función de activación de la neurona de la última capa tiene que ser "**linear**".

```
# define two sets of inputs
inputA = Input(shape=(32,))
inputB = Input(shape=(128,))

# the first branch operates on the first input
x = Dense(8, activation="relu")(inputA)
x = Dense(4, activation="relu")(x)
x = Model(inputs=inputA, outputs=x)

# the second branch opreates on the second input
y = Dense(64, activation="relu")(inputB)
y = Dense(32, activation="relu")(y)
y = Dense(4, activation="relu")(y)
y = Model(inputs=inputB, outputs=y)

# combine the output of the two branches
combined = concatenate([x.output, y.output])

# apply a FC layer and then a regression prediction on the
# combined outputs
z = Dense(2, activation="relu")(combined)
z = Dense(1, activation="linear")(z)

# our model will accept the inputs of the two branches and
# then output a single value
model = Model(inputs=[x.input, y.input], outputs=z)
```

Consejos para el desarrollo (arquitectura)

- La **rama** encargada de las **características numéricas y categóricas** será un **perceptron multicapa** de dos *hidden layers* con **número de neuronas descendente**.
- La **rama** encargada de extraer la **información visual** contenida en las **imágenes** será un **red neuronal convolucional** con un número de bloques a determinar definidos por un número de filtros ascendente.
- Dicha **rama convolucional** deberá acabar con un ***top model*** cuya **última** capa ***fully-connected*** deberá tener el **mismo número** de **neuronas** que la **última capa** de la **otra rama**.
- **Ambas ramas** serán **concatenadas** y llevadas **a una capa Dense** del **mismo número de neuronas** que la última de las capas de ambas ramas. Por último se establecerá la **capa de salida** con una **única neurona** y función de activación "**linear**".

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

CVBLAB

DEPARTAMENTO DE COMUNICACIONES

CENTRO FORMACIÓN PERMANENTE

Deep Learning aplicado al análisis de señales e imágenes

μPROYECTOS