SISTEMAS MULTIAGENTES

Grupo Planta

Entrega N° 02

INTEGRANTES DEL GRUPO

Apellido y Nombres	E-Mail	Porcentaje de Aporte
Bassi, Matías Lionel	mbassi@upe.edu.ar	25 %
Chimuris Giménez, Andrés Daniel	achimurisgimenez@frba.utn.edu.ar	25 %
Garrido, Nicolás Matías	garridonm@gmail.com	25 %
Pozzi, Gastón	gastonpozzi@gmail.com	25 %

Fecha de Presentación	11/10/2022

UTN.BA FACULTAD REGIONAL BUENOS AIRES

TP 02 - Plantas

Resumen. Implementación de Prototipo usando RNA Tradicionales

1. Introducción

En el marco del Trabajo Práctico Integrador, que contempla el desarrollo de un sistema multiagente, se ha seleccionado una sub-meta asociada al objetivo general para implementar un prototipo como prueba de concepto para intentar resolverla usando algún modelo de Redes Neuronales Artificiales Tradicionales (por ejemplo, RNA MLP BackPropagation). Para ello, a través de herramienta/s, framework/s y/o librería/s disponible/s, se ha llevado a cabo por lo menos 3 entrenamientos (exitosos o no) usando diferentes arquitecturas, parámetros y/o topologías, incluyendo el análisis de las métricas aplicadas para su prueba. Como resultado de esta actividad, el grupo entrega el presente informe que incluye:

- Sub-meta seleccionada del problema a resolver.
- Herramienta/s, framework/s y/o librería/s seleccionada/s para la implementación.
- Datos disponibles para entrenar el prototipo:
    - Una tabla con por lo menos 5 ejemplos de los a datos a ser utilizados
    - Descripción de las técnicas de preparación aplicadas sobre los datos disponibles (en caso de que haya sido necesario).
    - Descripción del método empleado para seleccionar los datos para la construcción y prueba del prototipo (al menos se requiere que 25% de los ejemplos para la prueba).
- Por cada entrenamiento se indica al menos:
    - El Modelo y la Arquitectura de RNA utilizada.
    - La topología de la RNA y valores de los parámetros utilizados en el entrenamiento.
    - El gráfico del progreso del Error General (Loss) obtenido durante el entrenamiento.
    - Las Métricas correspondientes obtenidas utilizando los datos de prueba.
    - Por lo menos 5 ejemplos de prueba del prototipo. Para ello, se debe utilizar una tabla que indique:
        - los datos de entrada ingresados,
        - la salida generada por la red,
        - la salida esperada para los datos de entrada (si está disponible), y
        - la comparación entre la salida esperada y la generada (ok o error).
- Análisis de los resultados obtenidos en la sección anterior.
- Problemas e inconvenientes encontrados durante la realización de esta entrega.
- Posibles cursos de acción para mejorar los resultados obtenidos justificando su viabilidad.

2. Submeta del problema a resolver

El desarrollo del primer agente el cual se encarga de indicar si la imagen procesada es o no una planta. En caso de que no sea una planta, se la descarta.

3. Herramienta/s, framework/s y/o librería/s seleccionada/s para la implementación.

Se ha utilizado, en línea con lo aprendido en el Seminario, las siguientes herramientas: Google Colab, Tensor Flow, Keras, Sklearn, matplotlib, panda, numpy, os y PIL.

4. Datos disponibles para entrenar el prototipo

Se dispone de un conjunto de imágenes extraídas desde distintas fuentes. Las mismas han sido clasificadas, manualmente, en dos clases “planta” y “no planta”. A su vez se seleccionan algunas imágenes para “Test” y otras para el entrenamiento en sí.

Tipo	Cantidad
Train - No Planta	37
Train - Planta	37
Test - No planta	12
Test - Planta	6

Las mismas pueden ser consultadas desde el siguiente link: Google Drive

Respecto al método empleado para seleccionar los datos para la construcción y prueba del prototipo, ha de mencionarse que la selección de imágenes para “entrenamiento” y para “test” se ha realizado manualmente. En dicho proceso se intentó buscar que exista diversidad de imágenes en los distintos conjuntos.

5. Entrenamientos realizados

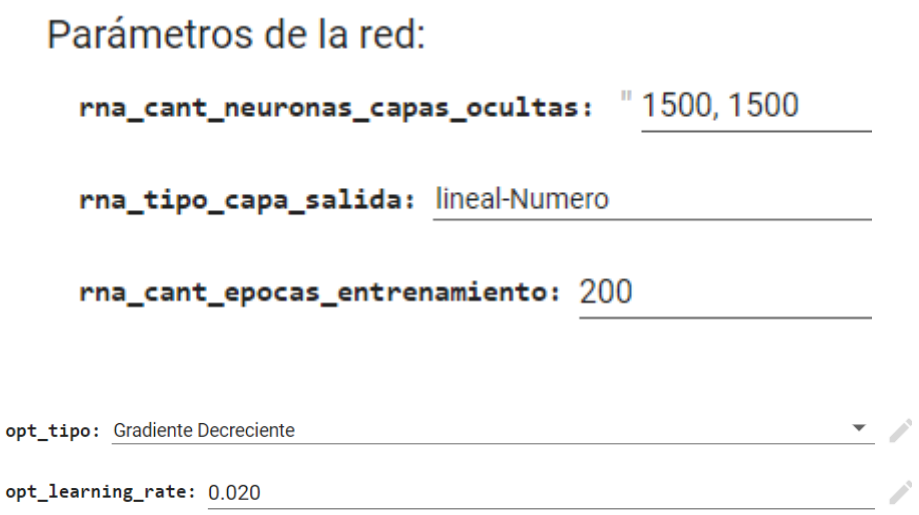
Se realizaron un total de 6 entrenamientos. En todos se ha utilizado una RNA MULTI-PERCEPTRÓN BACKPROPAGATION.

A continuación se encontrará el acceso al archivo notebook de cada entrenamiento:

#	Link
#1	https://drive.google.com/file/d/1tnuxeq3uoagYTDNSapoUpVRxzaF7_yFB/view?usp=sharing
#2	https://drive.google.com/file/d/1LhJV34rnkoP7TG171HG_y7OgllTEHbik/view?usp=sharing
#3	https://drive.google.com/file/d/1na2_Mj7wj2QO1St3FbzM_4Iap79vp5lS/view?usp=sharing
#4	https://drive.google.com/file/d/1wife7a1w5gQSVqstb4VeECkd-ltt2XkT/view?usp=sharing
#5	https://drive.google.com/file/d/1xQPBI6f0cNh8qRUNqjhB8FFIOKSnuM5h/view?usp=sharing
#6	https://drive.google.com/file/d/1_KBocjkubo6PwjxcoJug2lRljgAAnVtf/view?usp=sharing

5.1. Entrenamiento #1

- 2 capas de neuronas ocultas
    - 1500 neuronas en cada capa
- Sin capa Dropout
- Sin capa Batch Normalization
- Capa Salida: linealNumero
- 200 ciclos de iteración
- Propagación del Error mediante Gradiente Decreciente
- Tasa de Aprendizaje: 0,020.



Nos arrojo un error de “clase inválida”:

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-18-2ab0adc1b5ec> in <module>
     74 # prueba con los datos de prueba
     75 print("*** Resultados con datos de Entrenamiento: ")
---> 76 probarModelo(x_train, y_train, esDAimag_train, clases_map, mostrar_detalle_imagenes_entrenamiento)

<ipython-input-18-2ab0adc1b5ec> in probarModelo(x, y, esDAimag, clases_map, mostrarImagenes)
     25             ## determina clase predecida de acuerdo al umbral de clasificación
     26             idclPred = predClass[i][0]
---> 27             idclPredRnd = int(idclPred)
     28             if (idclPred - idclPredRnd)>0.5 and (idclPredRnd+1)<len(clases_map):
     29                     idclPredRnd = idclPredRnd + 1

ValueError: cannot convert float NaN to integer
```

5.2. Entrenamiento #2

Se modifica el entrenamiento #1. Se ajusta la capa de salida: Softmax-Multiclase. El resto de los parámetros se mantienen iguales.

- 2 capas de neuronas ocultas
    - 1500 neuronas en cada capa
- Sin capa Dropout
- Sin capa Batch Normalization
- Capa Salida: Softmax-Multiclase
- 200 ciclos de iteración
- Propagación del Error mediante Gradiente Decreciente
- Tasa de Aprendizaje: 0,020

.

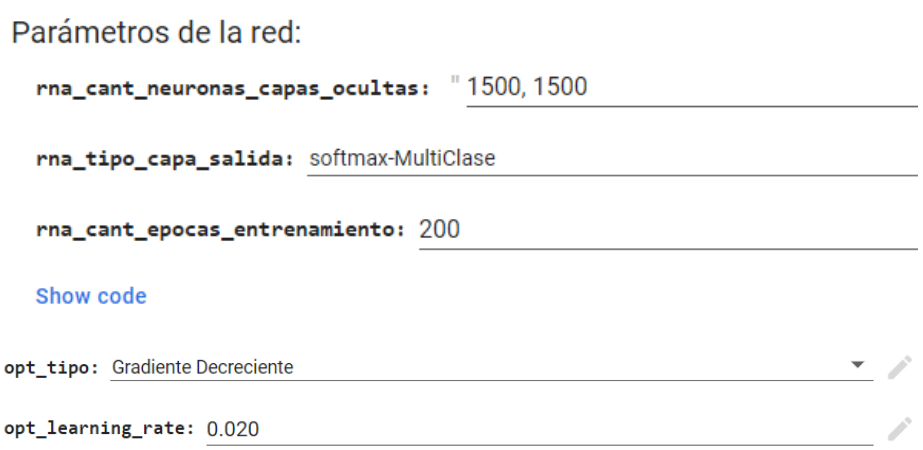


Gráfico del error del entrenamiento

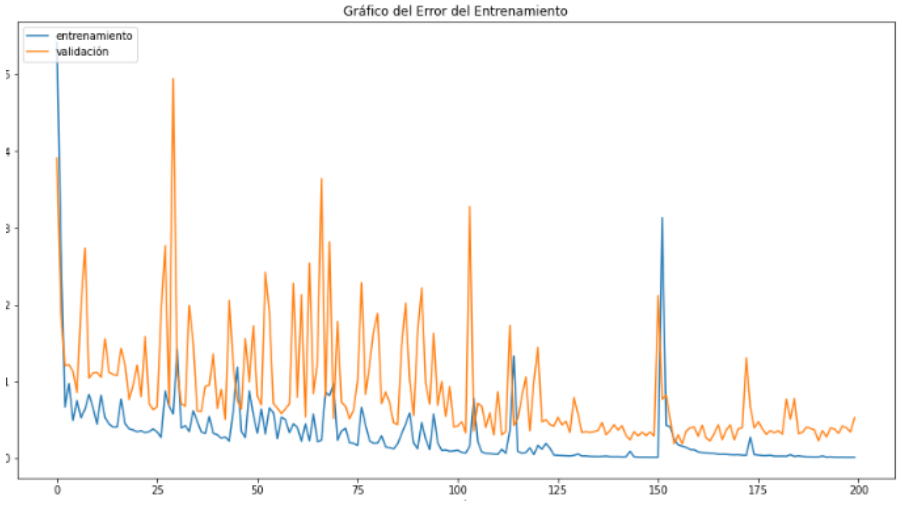
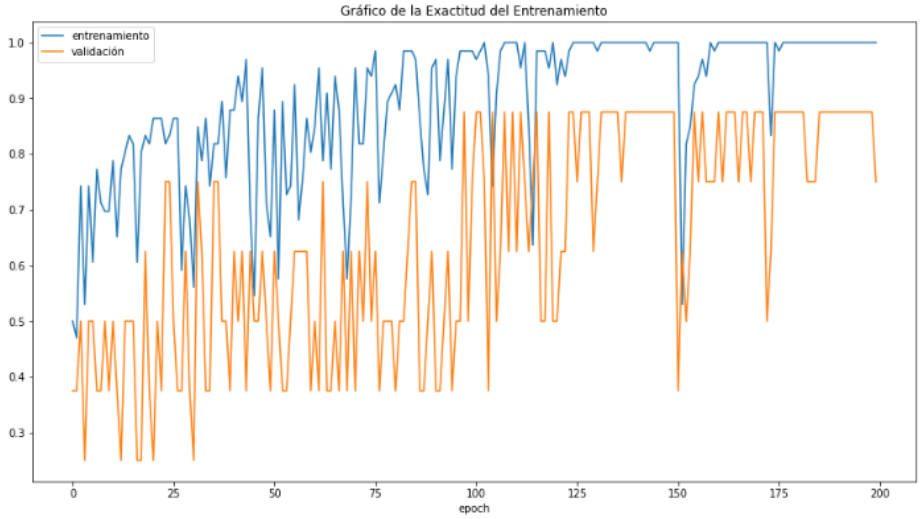


Gráfico del Error del Entrenamiento
entrenamiento
validación
0
25
50
75
100
125
150
175
200

Gráfico de exactitud



Datos de entrenamiento:

Reporte de Clasificación:

	precision	recall	f1-score	support
no_planta	1.00	0.95	0.97	37
planta	0.95	1.00	0.97	37
accuracy			0.97	74
macro avg	0.97	0.97	0.97	74
weighted avg	0.97	0.97	0.97	74

Matriz de Confusión (real / modelo):

	m:no_planta	m:planta
r:no_planta	35	2
r:planta	0	37

Datos de pruebas:

```
1/1 [==============================] - 0s 19ms/step - loss: 0.8435 - accuracy: 0.7222

>Evaluación del Modelo:
    - Error:  0.8434847593307495
    - Exactitud:  72.22222089767456



*** Resultados con datos de Prueba:

 Reporte de Clasificación:
              precision    recall  f1-score   support

   no_planta       0.82      0.75      0.78        12
      planta       0.57      0.67      0.62         6

    accuracy                           0.72        18
   macro avg       0.69      0.71      0.70        18
weighted avg       0.74      0.72      0.73        18


 Matriz de Confusión ( real / modelo ):
             m:no_planta  m:planta
r:no_planta            9         3
r:planta               2         4
```

5.3. Entrenamiento #3

Parámetros de la red:

rna_cant_neuronas_capas_ocultas: 1500, D, 1500

rna_tipo_capa_salida: softmax-MultiClase

rna_cant_epocas_entrenamiento: 200

Show code

opt_tipo: Gradiente Decreciente

opt_learning_rate: 0.020

Gráfico del error del entrenamiento

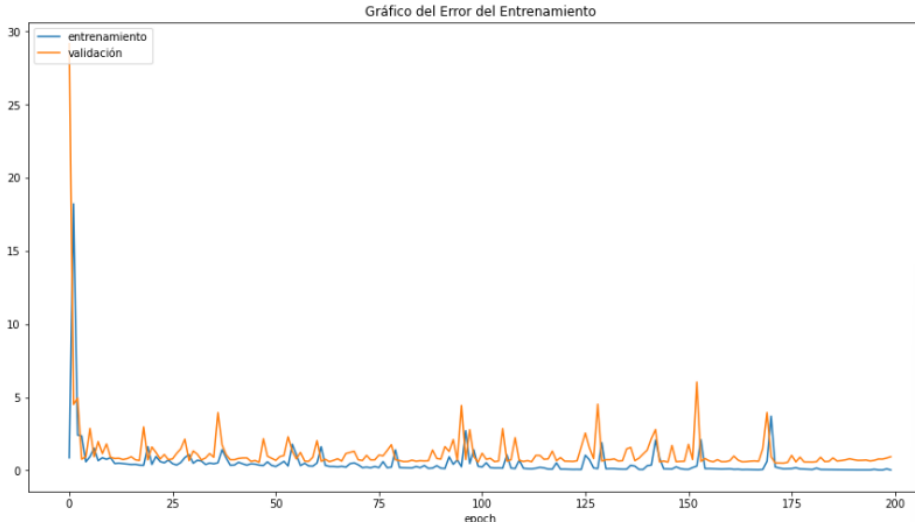
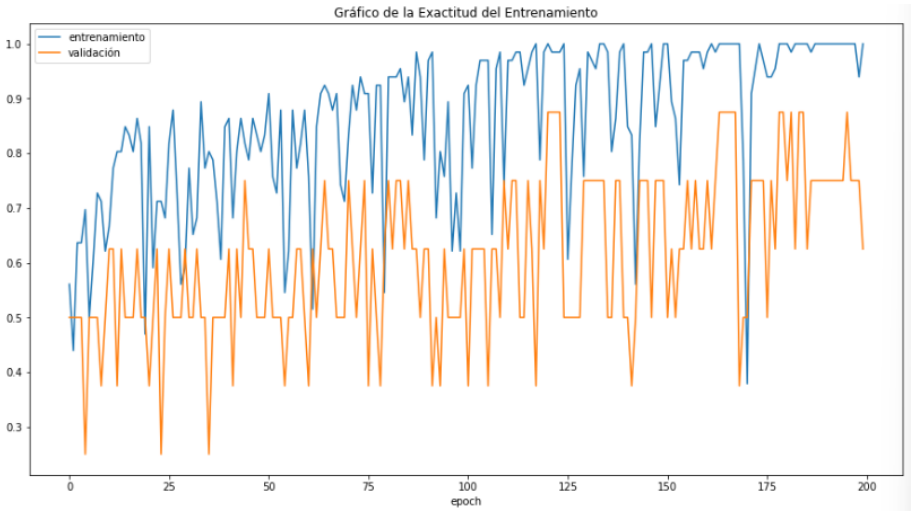


Gráfico de la Exactitud del Entrenamiento



Resultados del entrenamiento

*** Resultados con datos de Entrenamiento:

Reporte de Clasificación:

	precision	recall	f1-score	support
no_planta	1.00	0.92	0.96	37
planta	0.93	1.00	0.96	37
accuracy			0.96	74
macro avg	0.96	0.96	0.96	74
weighted avg	0.96	0.96	0.96	74

Matriz de Confusión (real / modelo):

	m:planta	m:no_planta
r:planta	37	0
r:no_planta	3	34

Resultados de la prueba

```
>Evaluación del Modelo:
    - Error:  0.86042320728302
    - Exactitud:  61.11111044883728



*** Resultados con datos de Prueba:

 Reporte de Clasificación:
              precision    recall  f1-score   support

   no_planta       0.78      0.58      0.67        12
      planta       0.44      0.67      0.53         6

    accuracy                           0.61        18
   macro avg       0.61      0.62      0.60        18
weighted avg       0.67      0.61      0.62        18


 Matriz de Confusión ( real / modelo ):
             m:planta  m:no_planta
r:planta            4            2
r:no_planta         5            7
```

5.4. Entrenamiento #4

Parámetros de la red:

rna_cant_neuronas_capas_ocultas: " 1500, D, 1500

rna_tipo_capa_salida: softmax-MultiClase

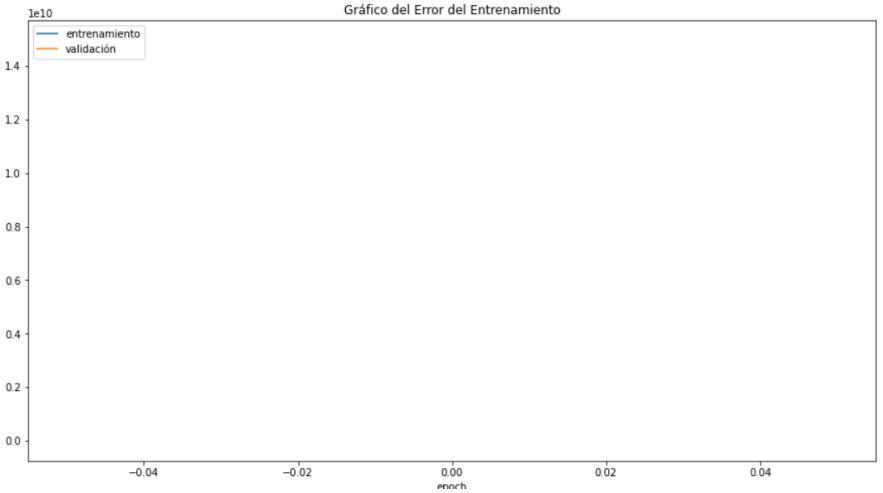
rna_cant_epocas_entrenamiento: 200

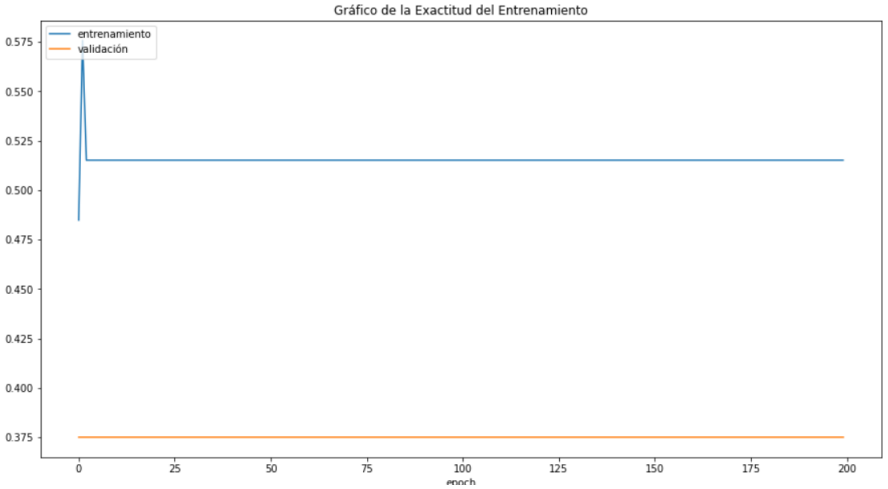
Show code

opt_tipo: Gradiente Decreciente

opt_learning_rate: 0.20

Se modificó alpha, se subió a 0,20. Estos son los resultados:





*** Resultados con datos de Entrenamiento:

Reporte de Clasificación:

	precision	recall	f1-score	support
no_planta	0.00	0.00	0.00	37
planta	0.50	1.00	0.67	37
accuracy			0.50	74
macro avg	0.25	0.50	0.33	74
weighted avg	0.25	0.50	0.33	74

Matriz de Confusión (real / modelo):

	m:planta	m:no_planta
r:planta	37	0
r:no_planta	37	0

```
>Evaluación del Modelo:
    - Error:  nan
    - Exactitud:  33.33333432674408



*** Resultados con datos de Prueba:

 Reporte de Clasificación:
              precision    recall  f1-score   support

   no_planta       0.00      0.00      0.00        12
      planta       0.33      1.00      0.50         6

    accuracy                           0.33        18
   macro avg       0.17      0.50      0.25        18
weighted avg       0.11      0.33      0.17        18


 Matriz de Confusión ( real / modelo ):
             m:planta  m:no_planta
r:planta            6            0
r:no_planta        12            0
```

5.5. Entrenamiento #5

Parámetros de la red:

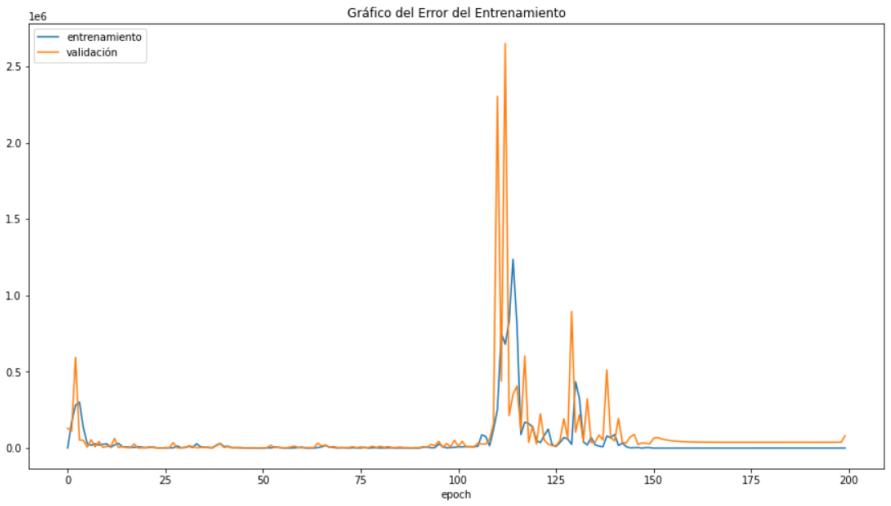
rna_cant_neuronas_capas_ocultas: 1500, D, 1500

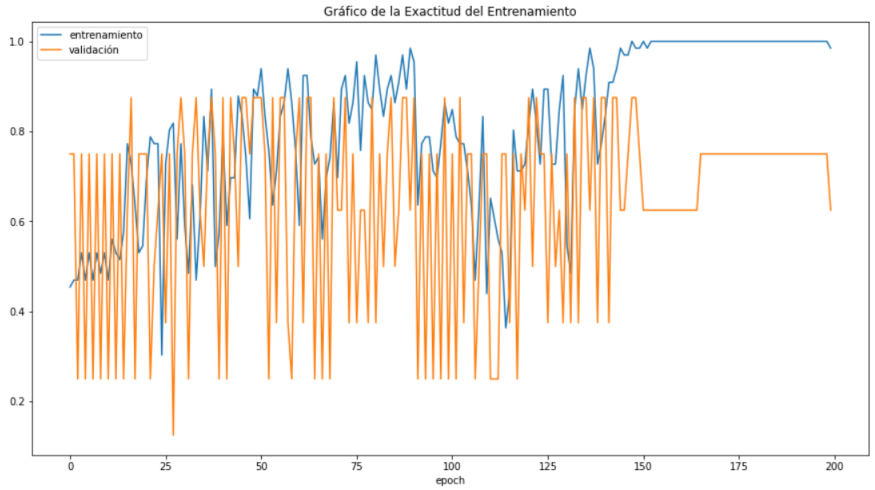
rna_tipo_capa_salida: softmax-MultiClase

rna_cant_epocas_entrenamiento: 200

opt_tipo: Adam

opt_learning_rate: 0.050





*** Resultados con datos de Entrenamiento:

Reporte de Clasificación:

	precision	recall	f1-score	support
no_planta	0.95	0.97	0.96	37
planta	0.97	0.95	0.96	37
accuracy			0.96	74
macro avg	0.96	0.96	0.96	74
weighted avg	0.96	0.96	0.96	74

Matriz de Confusión (real / modelo):

	m:planta	m:no_planta
r:planta	35	2
r:no_planta	1	36

```
>Evaluación del Modelo:
    - Error:  135663.421875
    - Exactitud:  72.22222089767456



*** Resultados con datos de Prueba:

 Reporte de Clasificación:
              precision    recall  f1-score   support

   no_planta       0.82      0.75      0.78        12
      planta       0.57      0.67      0.62         6

    accuracy                           0.72        18
   macro avg       0.69      0.71      0.70        18
weighted avg       0.74      0.72      0.73        18


 Matriz de Confusión ( real / modelo ):
            m:planta  m:no_planta
r:planta           4            2
r:no_planta        3            9
```

5.6. Entrenamiento #6

Parámetros de la red:

rna_cant_neuronas_capas_ocultas: 1500, 500, D, 1000, D, 250

rna_tipo_capa_salida: softmax-MultiClase

rna_cant_epocas_entrenamiento: 200

opt_tipo: Adam

opt_learning_rate: 0.050

Gráfico del Error del entrenamiento

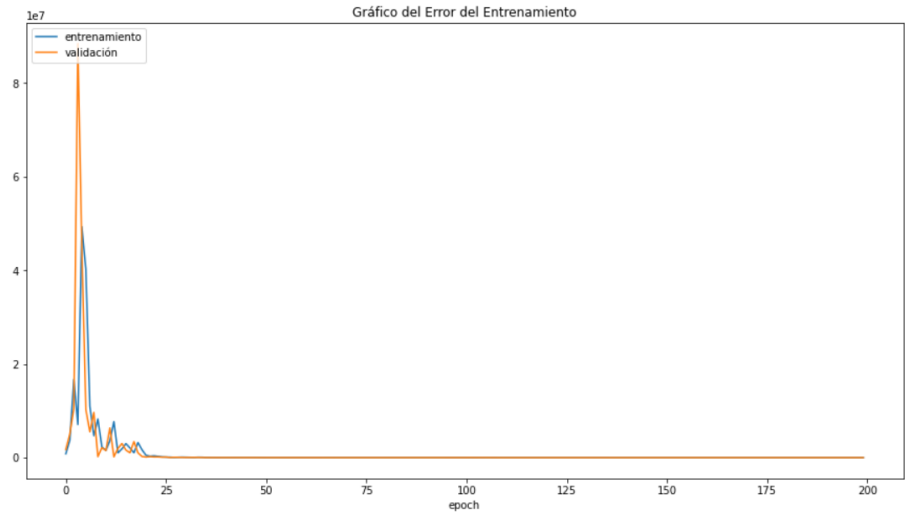
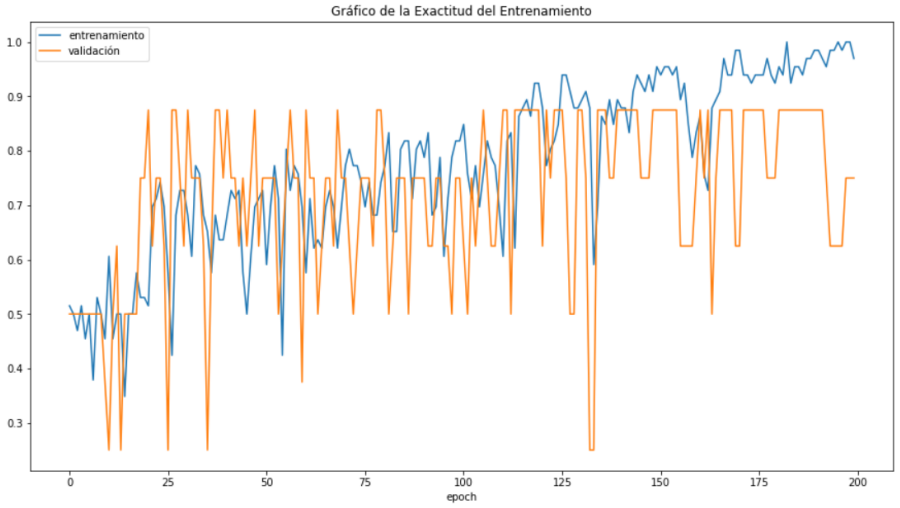


Gráfico de la exactitud del entrenamiento




```
*** Resultados con datos de Entrenamiento:

 Reporte de Clasificación:
              precision    recall  f1-score   support

   no_planta       0.97      0.97      0.97        37
      planta       0.97      0.97      0.97        37

    accuracy                           0.97        74
   macro avg       0.97      0.97      0.97        74
weighted avg       0.97      0.97      0.97        74


 Matriz de Confusión ( real / modelo ):
             m:planta  m:no_planta
r:planta           36            1
r:no_planta         1           36
```

```
>Evaluación del Modelo:
    - Error:  2102.89794921875
    - Exactitud:  83.33333134651184



*** Resultados con datos de Prueba:

 Reporte de Clasificación:
              precision    recall  f1-score   support

   no_planta       0.91      0.83      0.87        12
      planta       0.71      0.83      0.77         6

    accuracy                           0.83        18
   macro avg       0.81      0.83      0.82        18
weighted avg       0.84      0.83      0.84        18


 Matriz de Confusión ( real / modelo ):
             m:planta  m:no_planta
r:planta            5            1
r:no_planta         2           10
```

6. Análisis de los resultados obtenidos

En el primer entrenamiento, nos dio un error por clasificación 1. Por lo tanto, no se pudo realizar un análisis.

En el segundo entrenamiento: obtuvimos resultados alentadores. Sin embargo, cuando se testeo/probó, los resultados NO fueron los esperados.

Debido a que en el segundo entrenamiento obtuvimos buenos resultados en la etapa de entrenamiento y malos en la etapa de test, nos percatamos de la necesidad de contar con una capa DropOut para evitar el sobre-entrenamiento. Otra posibilidad hubiera sido reducir el número de ciclos de entrenamiento.

Entonces, en el tercer modelo, se agregó una capa de DropOut porque se notó que para los datos de entrenamiento los resultados eran muy buenos y no para los de prueba. La red estaría “sobreentrenada”.

En el cuarto modelo, subimos la tasa de aprendizaje para evaluar el comportamiento de la red y analizar cómo se comporta con los datos de prueba. Nos dimos cuenta que habíamos hecho los entrenamientos anteriores con valores de “alpha" no recomendados.

Como el cuarto entrenamiento tuvo resultados muy malos, se vuelve a modificar, para el quinto entrenamiento, el learning rate a un valor (recomendado) de 0,05. Además, se modifica la propagación del error seleccionando la opción “Adam”.

Por último, se propone una topología más compleja para la 6ta. Agregando varias capas dos capas DropOut, y 4 capas ocultas. A continuación se detalla la topología:

```
. Modelo creado con  8  capas:
  Model: "RNA"
  _________________________________________________________________
  Layer (type)                 Output Shape              Param #
  =================================================================
  input_img (InputLayer)       [(None, 3072)]            0

  hidd_1 (Dense)               (None, 1500)              4609500

  hidd_2 (Dense)               (None, 500)               750500

  dropout (Dropout)            (None, 500)               0

  hidd_3 (Dense)               (None, 1000)              501000

  dropout_1 (Dropout)          (None, 1000)              0

  hidd_4 (Dense)               (None, 250)               250250

  output (Dense)               (None, 2)                 502

  =================================================================
```

7. Problemas e inconvenientes encontrados durante la realización de esta entrega.

Durante el desarrollo de la presente entrega, hemos encontrado un inconveniente técnico. El mismo ha sucedido en el marco del entrenamiento #1. No nos ha sido posible completar dicho entrenamiento, debido a que el error nos lo imposibilitó. Por último, cabe destacar que no hemos dedicado un esfuerzo sustancial en la búsqueda de la solución pertinente ni en el entendimiento del problema.

8. Posibles cursos de acción para mejorar los resultados obtenidos justificando su viabilidad.

Para el presente TP se partió de la premisa de utilizar únicamente RNA Tradicionales. Sin embargo, debido a los malos resultados obtenidos, en los sucesivos entrenamientos hemos incorporado capas DropOut (para evitar el sobre-entrenamiento) y capa de salida de tipo SoftMax Multiclase. Luego, con significativas mejoras en la topología y el uso de estas capas no tradicionales, hemos conseguido resultados aceptables en el entrenamiento #6.

A partir de la experiencia obtenida al realizar el presente TP, consideramos que el agente que define si la imagen de un objeto puede clasificarse como planta o no presenta un resultado SATISFACTORIO. No obstante, consideramos que podríamos realizar más entrenamientos a los fines de mejorar los resultados.