

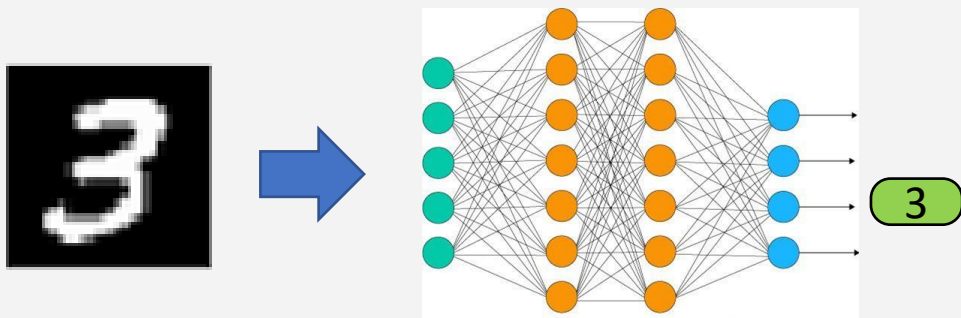
# Clasificación de Imágenes

The background of the slide is a photograph of a clear night sky. The Milky Way galaxy is visible as a bright, hazy band of light stretching across the upper half of the frame. Numerous individual stars are scattered throughout the dark blue sky. In the lower portion of the image, the dark, silhouetted branches of evergreen trees are visible, framing the bottom and right sides of the scene.

# Clasificación de imágenes

Para crear un modelo que permita clasificar imágenes es necesario tener en cuenta algunas consideraciones:

- Las imágenes tienen los píxeles en formato matricial. Un modelo neuronal no acepta esto. Es necesario “aplanar” la imagen.
- Si es una imagen color, la cantidad de píxeles se triplica al tener los 3 canales.



# Clasificación de imágenes

Imagen original

0	1	2
3	4	5
6	7	8

Aplanar

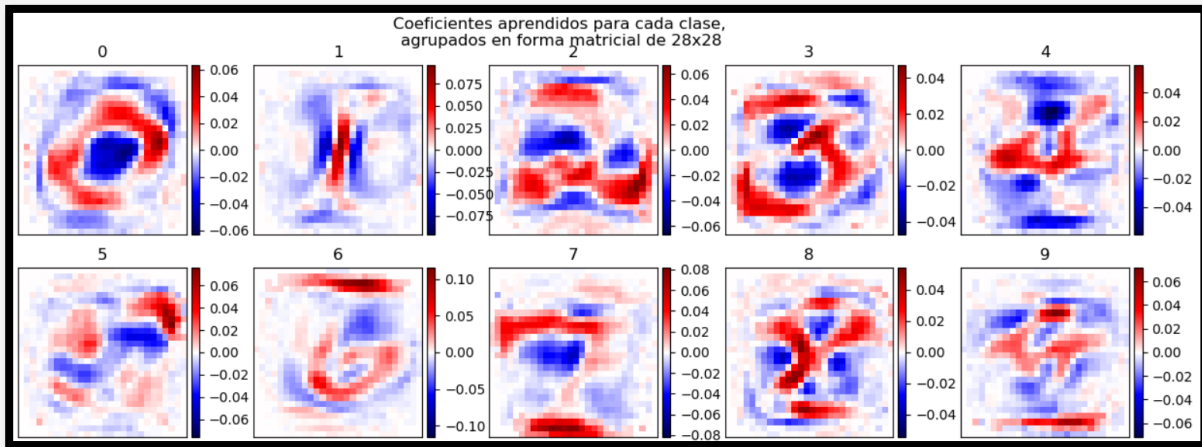
Imagen aplanada

0	1	2	3	4	5	6	7	8
---	---	---	---	---	---	---	---	---

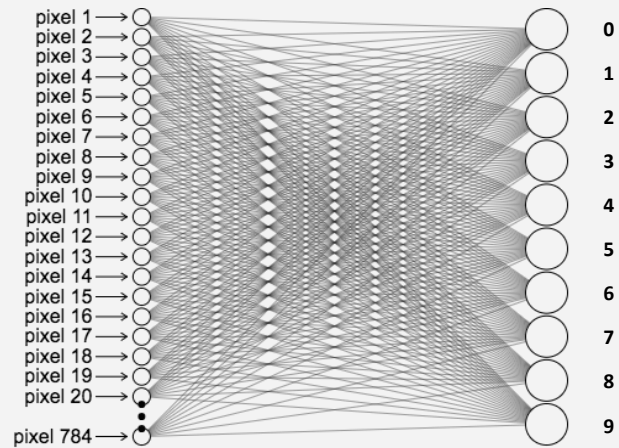
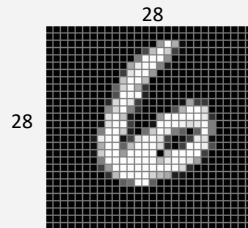
```
# Reshape de una imagen individual
img = np.reshape(img, (img.shape[0]* img.shape[1]))
# Reshape de todo el dataset
X = np.reshape(X, (X.shape[0], X.shape[1]*X.shape[2]))
```

```
# En Keras se agrega una capa “flatten” al comienzo para que aplane la imagen.
model.add(Flatten(input_shape= X_train[0].shape))
```

# Clasificación de imágenes - MNIST



Regresión Logística





# Clasificación de imágenes – CIFAR10

CIFAR10 es una base de datos mucho más realista que contiene decenas de miles de imágenes con diferentes objetos como animales y vehículos. Cada imagen está codificada en 32x32 píxeles con 3 canales de color.



# Clasificación de imágenes – CIFAR10

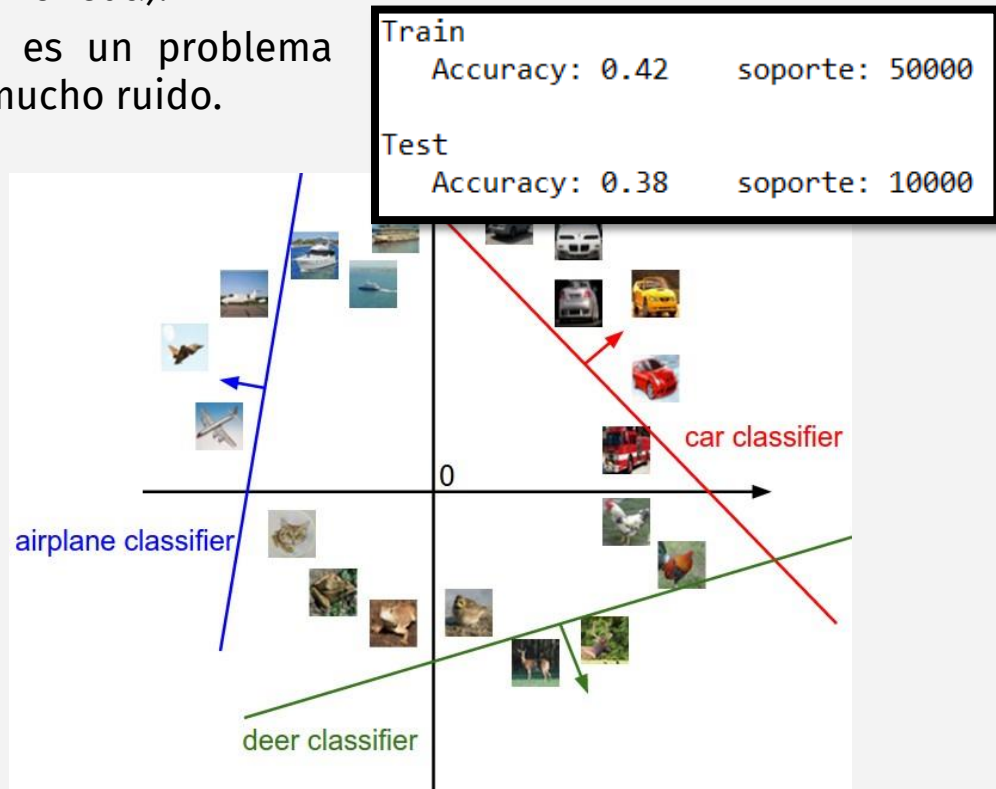
**38%** de Acc es mejor que  $1/10 = 10\%$  (tirar una moneda).

Estos resultados son esperables ya que no es un problema simple. Las imágenes son muy distintas y con mucho ruido.

Confusion matrix

airplane	312	54	21	4	9	119	17	103	259	102
automobile	28	487	14	9	10	72	24	78	76	202
bird	70	51	176	11	56	254	114	170	63	35
cat	13	63	53	51	26	445	105	112	46	86
deer	32	27	97	10	163	244	152	205	33	37
dog	14	49	50	27	27	551	54	135	55	38
frog	12	51	52	19	51	235	431	88	22	39
horse	19	51	31	7	31	155	27	554	37	88
ship	58	73	5	2	1	97	6	40	581	137
truck	36	176	10	5	8	51	39	71	77	527

airplane automobile bird cat deer dog frog horse ship truck

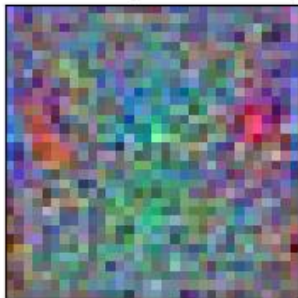


# Clasificación de imágenes – CIFAR10

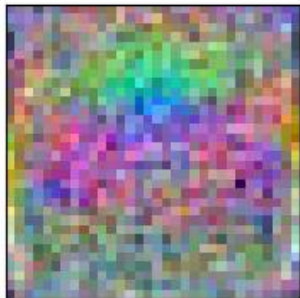
Vector de pesos aprendidos por cada clase (graficados en forma de matriz).

```
w_r= np.reshape(w, (32,32,3,10))
```

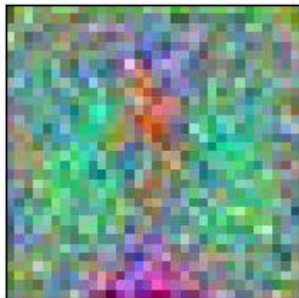
airplane



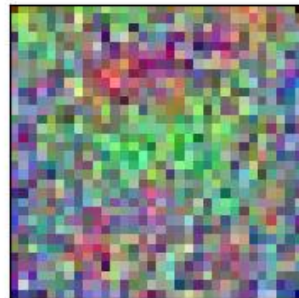
automobile



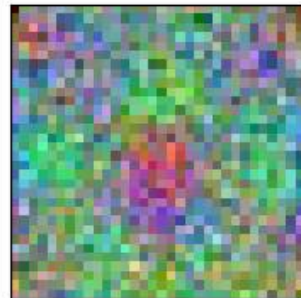
bird



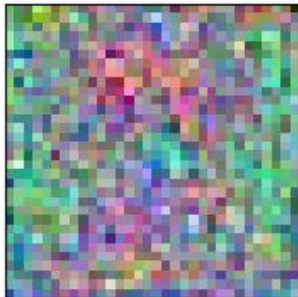
cat



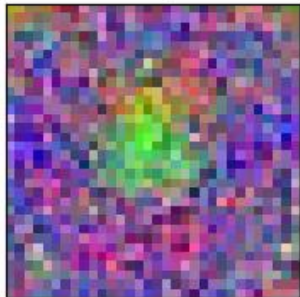
deer



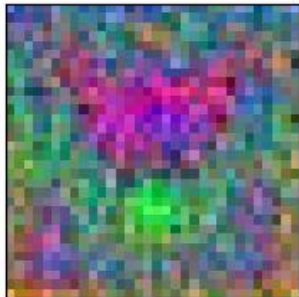
dog



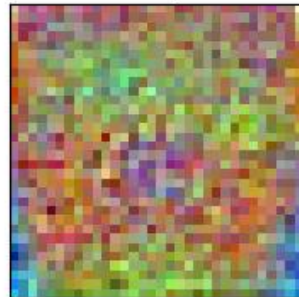
frog



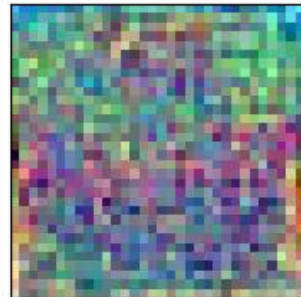
horse



ship

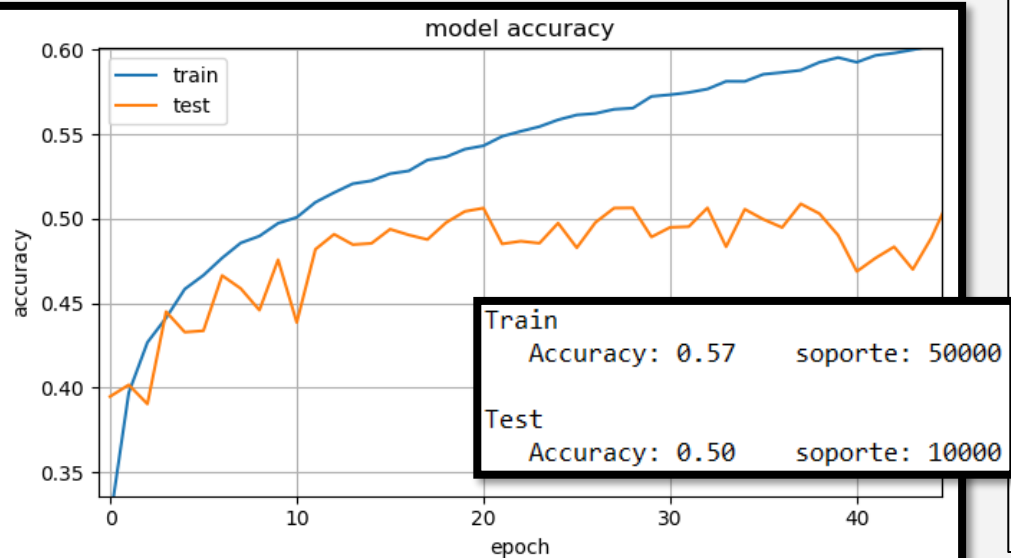


truck



# Clasificación de imágenes – CIFAR10

Al agregar una capa oculta a la red elevamos el Acc a 50% (para el testing set). Cuantas más capas ocultas, corremos riesgo de caer en Overfitting.



Confusion matrix

airplane	497	7	156	21	15	23	27	59	155	40
automobile	45	400	37	33	10	34	26	38	122	255
bird	50	8	509	65	77	78	105	79	14	15
cat	19	1	157	288	53	198	135	89	25	35
deer	36	1	221	48	353	45	148	112	25	11
dog	10	2	139	197	40	391	95	95	22	9
frog	3	4	83	62	86	44	652	39	13	14
horse	27	5	101	43	51	75	31	629	14	24
ship	84	19	48	22	24	39	12	24	664	64
truck	39	67	40	40	12	26	29	88	91	568
	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck



## Arquitectura

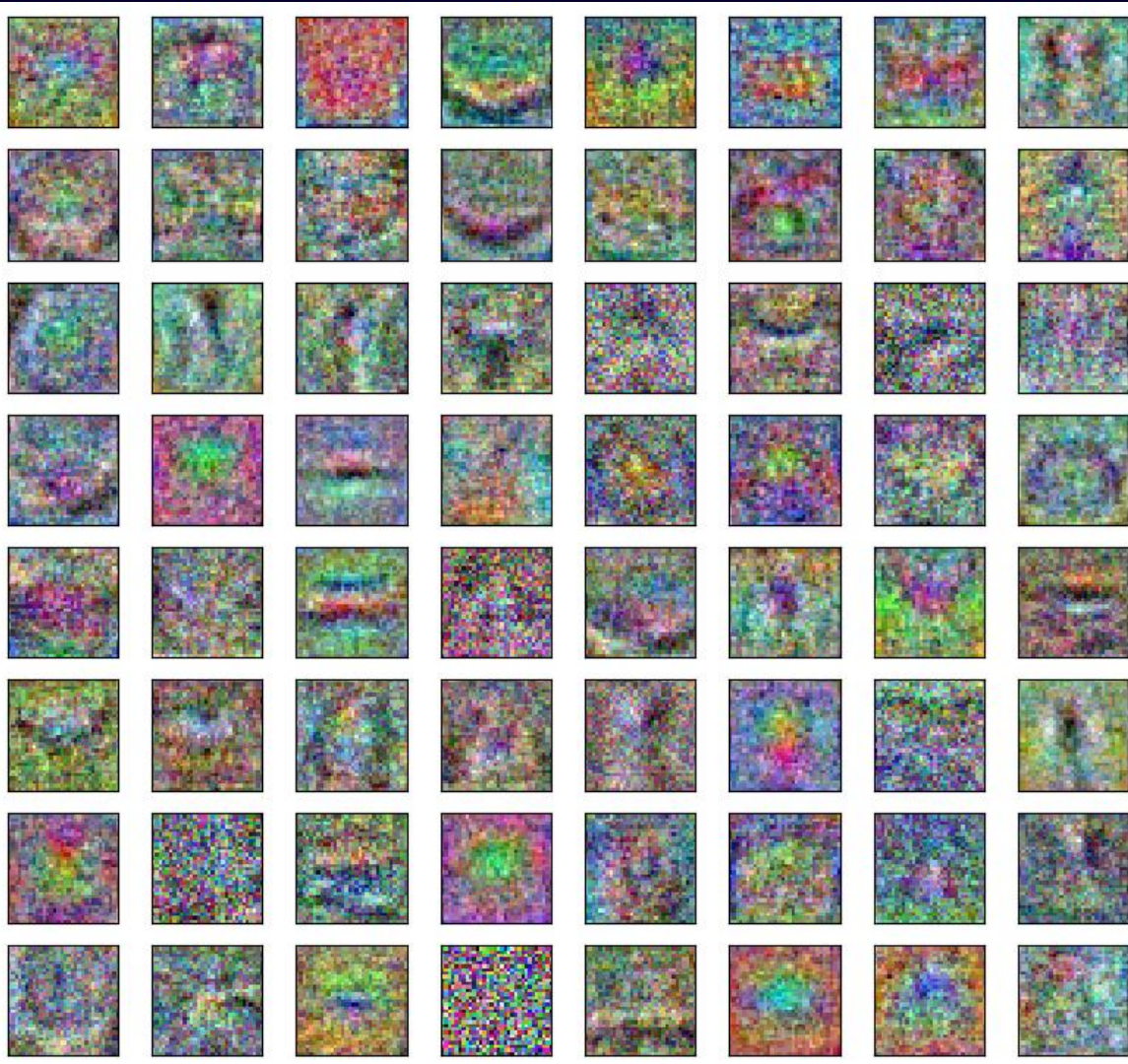
Flatten()

Dense(64, 'Relu')

Dense(10, 'softmax')

Pesos de las **64** neuronas de la capa oculta.

Notar que pareciera que algunas neuronas “encuentran” ciertos patrones en las imágenes.



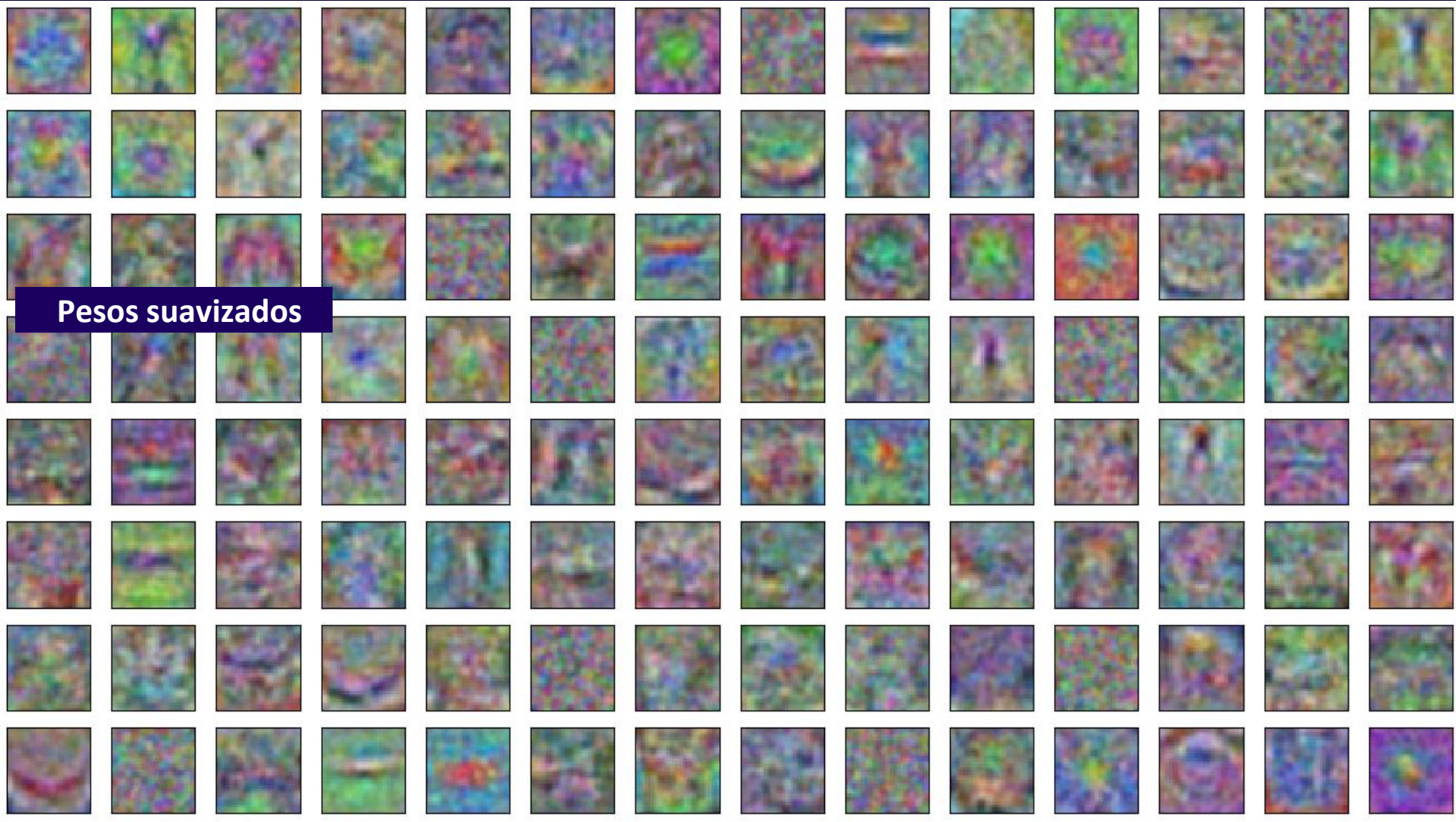


112 neuronas ocultas





Pesos suavizados



# Matriz de confusión

	predicted airplane predicted automobile predicted bird predicted cat predicted deer predicted dog predicted frog predicted horse predicted ship predicted truck										
airplane	851	15	19	10	23	4	6	15	43	19	85%
automobile	8	891	1	4	2	5	9	1	15	39	91%
bird	63	2	695	34	76	52	57	41	7	5	67%
cat	19	4	44	570	59	175	88	38	8	11	56%
deer	16	1	44	32	774	28	38	57	4	5	77%
dog	6	2	33	108	28	680	27	46	2	5	73%
frog	8	8	37	35	34	24	866	7	6	4	84%
horse	8	3	22	28	41	38	6	847	3	5	85%
ship	80	35	9	13	7	3	4	2	857	15	84%
truck	26	55	5	10	3	3	6	7	12	854	87%
precision	78%	88%	76%	68%	74%	67%	78%	80%	90%	89%	accuracy 79%

automobile misclassified as frog



En el siguiente link podrán encontrar una visualización del entrenamiento de redes neuronales para las dos bases de datos (MNIST y CIFAR10), tanto con redes neuronales feed-forward como con convolucionales (próxima clase).

[https://ml4a.github.io/demos/confusion\\_mnist/](https://ml4a.github.io/demos/confusion_mnist/)