

Entrenamiento de modelos de aprendizaje profundo mediante autosupervisión

Rubén Ezequiel Torti López

Facultad de Matemática, Astronomía, Física y Computación
Universidad Nacional de Córdoba

Agosto 2017

Aprendizaje automático

Explora algoritmos cuyo objetivo es identificar patrones en conjuntos de datos.

Aprendizaje automático

Explora algoritmos cuyo objetivo es identificar patrones en conjuntos de datos.

Supera el enfoque clásico de instrucciones estáticas, tomando decisiones basadas en datos.

Aprendizaje automático

Explora algoritmos cuyo objetivo es identificar patrones en conjuntos de datos.

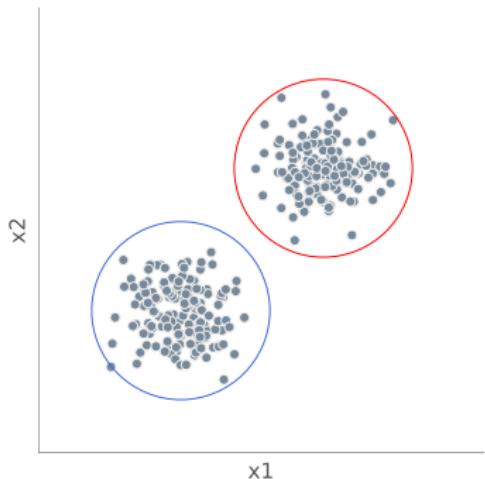
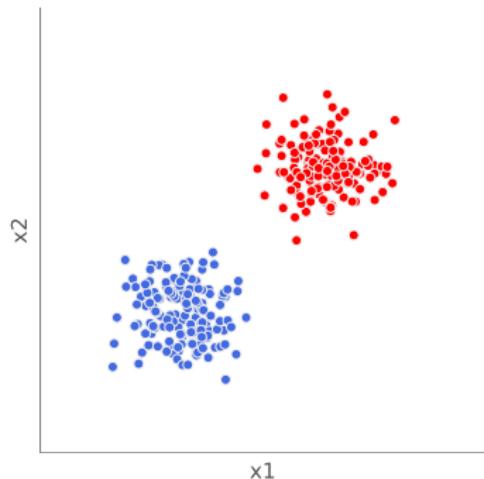
Supera el enfoque clásico de instrucciones estáticas, tomando decisiones basadas en datos.

Tareas demasiado complejas para programarse directamente:

Ejemplos

- Detección y verificación de caras
- Autos que se manejan solos
- Reconocimiento del habla
- Diagnósticos médicos

Aprendizaje Supervisado - No Supervisado



Visión por Computadoras

Busca describir y reconstruir propiedades del entorno a partir de imágenes.

Visión por Computadoras

Busca describir y reconstruir propiedades del entorno a partir de imágenes.

Problema inverso.

Visión por Computadoras

Busca describir y reconstruir propiedades del entorno a partir de imágenes.

Problema inverso.

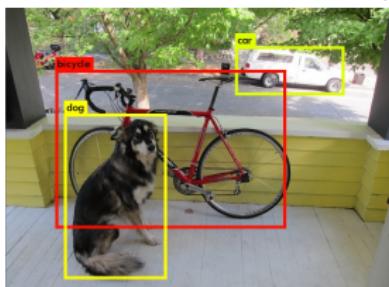
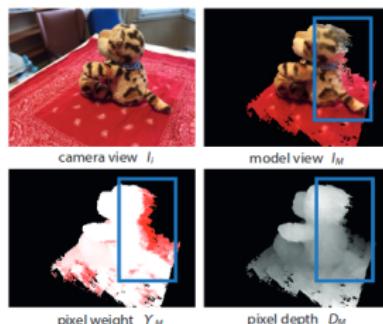
Es necesario recurrir a modelos físicos y/o probabilísticos

Visión por Computadoras

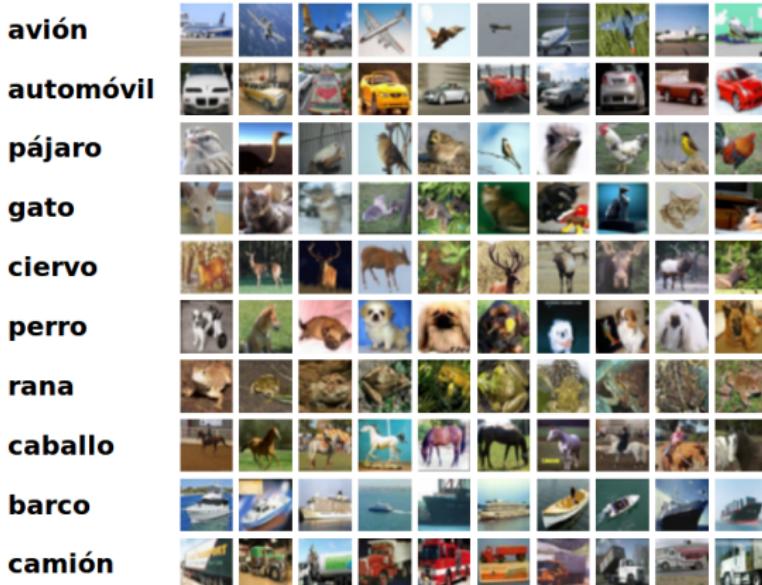
Busca describir y reconstruir propiedades del entorno a partir de imágenes.

Problema inverso.

Es necesario recurrir a modelos físicos y/o probabilísticos



Clasificación de imágenes



Representación de imágenes

¿Qué es una imagen?



```
[[[ 40,  36,  33],  
  [ 42,  38,  37],  
  [ 42,  38,  37],  
  ...,  
  [113, 115, 138],  
  [115, 117, 140],  
  [115, 117, 138]],  
  
 [[ 42,  38,  37],  
  [ 41,  37,  36],  
  [ 42,  36,  36],  
  ...,  
  [114, 116, 137],  
  [113, 114, 135],  
  [113, 114, 135]],  
  
 [ 42,  38,  37]
```

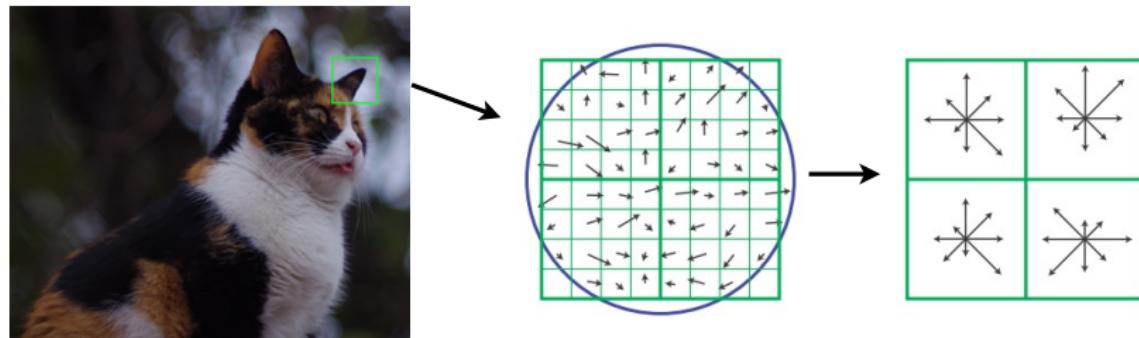
15 42 38 37

Representación de imágenes

Extracción de información de la imagen.

Dependiente del dominio del problema.

- Textura
- Color
- Descriptores locales



Podemos obtener representaciones de imágenes, $x \in \mathbb{R}^D$

Clasificación de imágenes

Conjunto de entrenamiento: $x_i \in \mathcal{R}^D$ Anotaciones asociadas: $y_i = 1 \cdots K$

$$f(x_i, W) = y_i \in 1 \cdots K \quad (1)$$

Clasificación de imágenes

Conjunto de entrenamiento: $x_i \in \mathcal{R}^D$ Anotaciones asociadas: $y_i = 1 \cdots K$

$$f(x_i, W) = y_i \in 1 \cdots K \quad (1)$$

- SVM
- Árboles de decisión
- *Bag of Words*

Entrenamiento

¿Cómo optimizar los parámetros W ?

$$f(x_i, W) = y_i \in 1 \cdots K \quad (2)$$

Entrenamiento

¿Cómo optimizar los parámetros W ?

$$f(x_i, W) = y_i \in 1 \cdots K \quad (2)$$

Proceso iterativo:

- Definir función de costo
- Minimizar la función de costo

Función de Costo

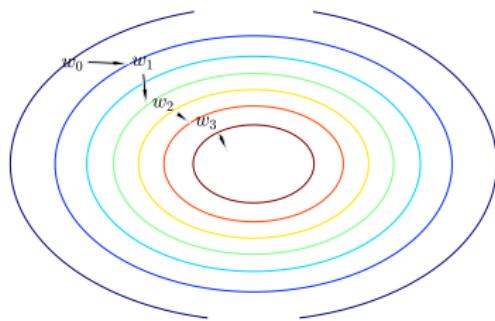
Define un criterio de optimalidad que nos ayuda a cuantificar cómo está actuando nuestro clasificador.

$$\mathcal{L}(W) = \frac{1}{m} \sum_i^m L(f(x_i; W), y_i) \quad (3)$$

Descenso de Gradiente Estocástico

Actualización de los pesos mediante el cómputo del costo en *mini-batches*

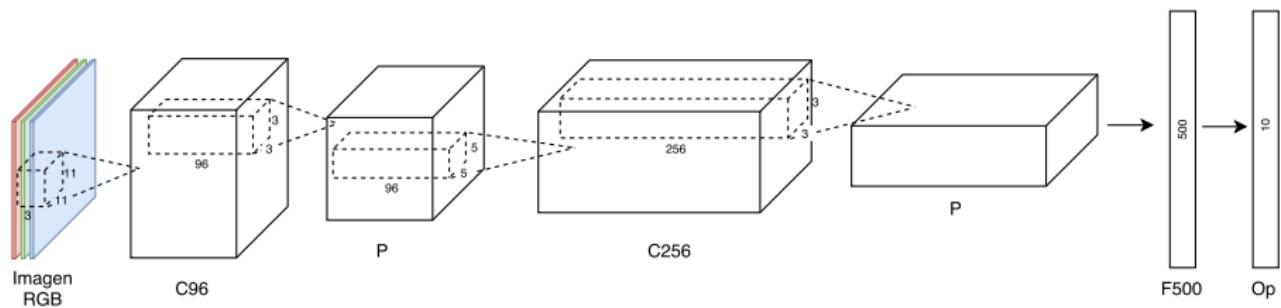
$$W_{n+1} = W_n - \epsilon \frac{1}{m} \sum_i^m \nabla_{W_n} L(f(x_i; W_n), y_i) \quad (4)$$



Clasificación de imágenes

Todos ampliamente superados por CNN

Pueden aprender representaciones Y el modelo .

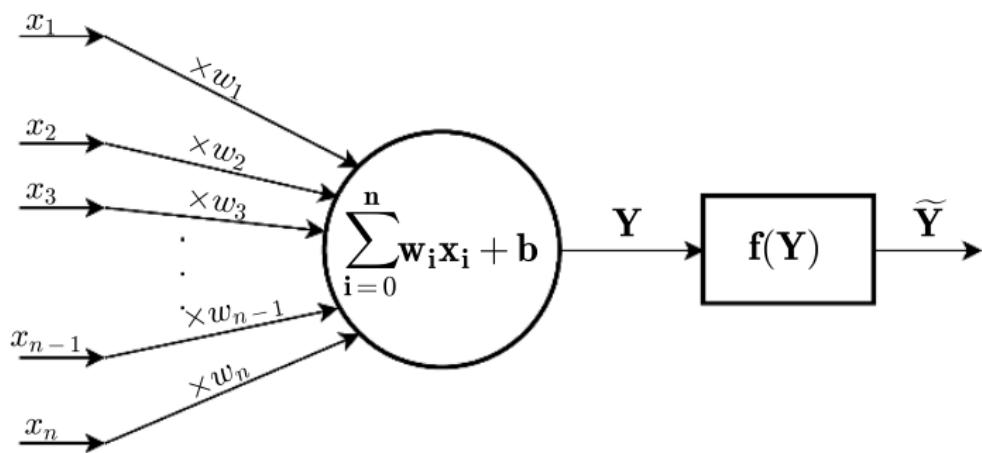


Redes Neuronales

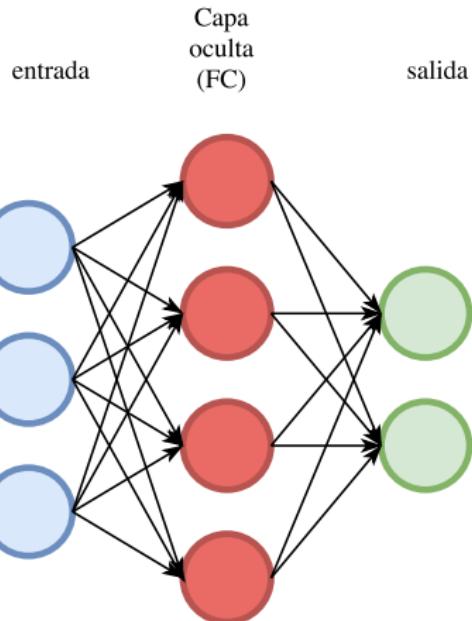
Unidades (neuronas) conectadas entre sí.

Típicamente organizadas en capas.

Neuronas de una capa interactúan con neuronas de la capa anterior.



Redes Neuronales



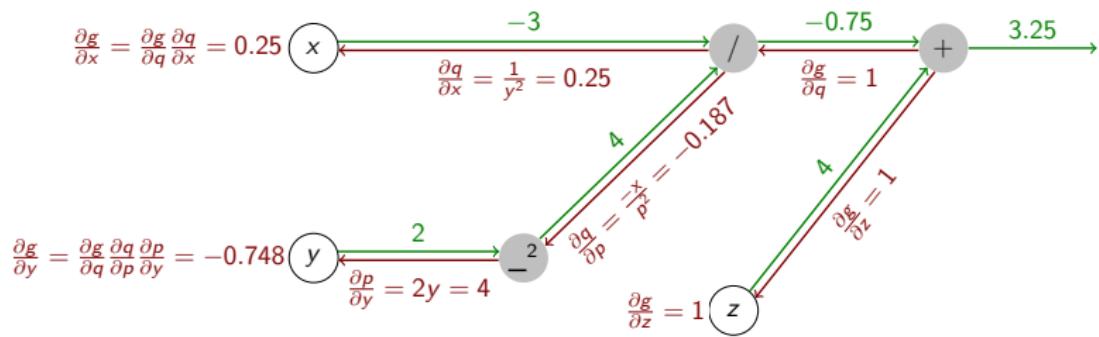
Optimización: Retropropagación

Algoritmo utilizado para computar eficientemente el gradiente.

- ① *Forward pass:* cómputo del error
- ② *Backward pass:* propagación del error a cada parámetro

Aplicación recursiva de la *regla de la cadena*:

$$g(x, y, z) = \frac{x}{y^2} + z = \frac{x}{p} + z = q + z \quad (5)$$



Redes Convolucionales

Imagen como una función $f : \mathbb{R}^2 \rightarrow \mathbb{R}^n$

Un *operador* toma una o más imágenes de entrada y produce una imagen de salida.

Operador de píxeles:

$$g(i, j) = h(f(i, j)) \quad (6)$$

filtro lineal: operador local

Redes Convolucionales

| | | | | | | | |
|----|----|----|-----|-----|-----|-----|-----|
| 45 | 60 | 98 | 127 | 132 | 133 | 137 | 133 |
| 46 | 65 | 98 | 123 | 126 | 128 | 131 | 133 |
| 47 | 65 | 96 | 115 | 119 | 123 | 135 | 137 |
| 47 | 63 | 91 | 107 | 113 | 122 | 138 | 134 |
| 50 | 59 | 80 | 97 | 110 | 123 | 133 | 134 |
| 49 | 53 | 68 | 83 | 97 | 113 | 128 | 133 |
| 50 | 50 | 58 | 70 | 84 | 102 | 116 | 126 |
| 50 | 50 | 52 | 58 | 69 | 86 | 101 | 120 |

$f(i, j)$

*

| | | |
|-----|-----|-----|
| 0.1 | 0.1 | 0.1 |
| 0.1 | 0.2 | 0.1 |
| 0.1 | 0.1 | 0.1 |

$h(i, j)$

=

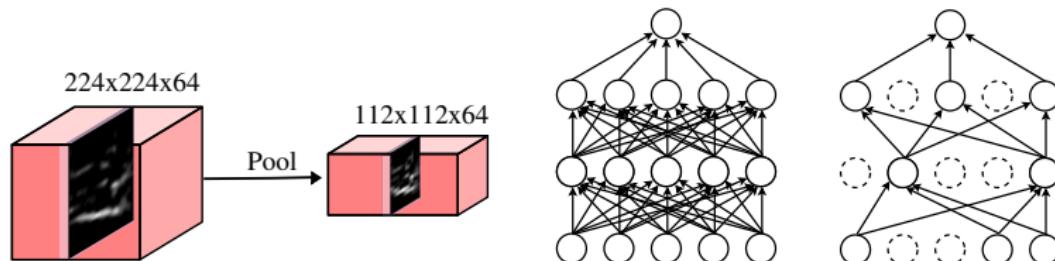
| | | | | | |
|----|----|-----|-----|-----|-----|
| 69 | 95 | 116 | 125 | 129 | 132 |
| 68 | 92 | 110 | 120 | 126 | 132 |
| 66 | 86 | 104 | 114 | 124 | 132 |
| 62 | 78 | 94 | 108 | 120 | 129 |
| 57 | 69 | 83 | 98 | 112 | 124 |
| 53 | 60 | 71 | 85 | 100 | 114 |

$g(i, j)$

Redes Convolucionales

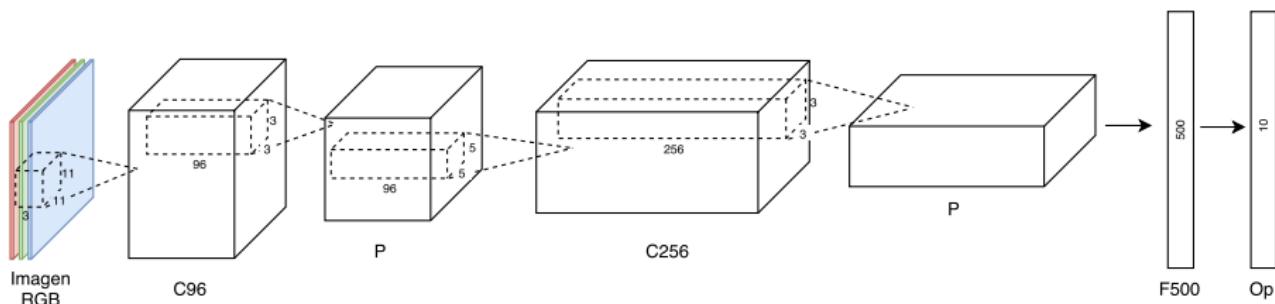
Convolución: localidad, reducción de parámetros, invarianza translacional
Capas:

- Convolucional
- Completamente conectada (FC)
- *Pooling*. Usualmente *MAX Pooling*
- *Dropout*, ReLU, Softmax



Redes Convolucionales

Ejemplo: C96-P-C256-F500-Op.



Luego de cada capa convolucional se utilizan unidades ReLU y luego de cada capa completamente conectada se agrega una capa de *Dropout* para prevenir sobreajuste de los datos.

(Algunos) problemas de Redes Convolucionales

- Requieren grandes conjuntos de datos
- Conjuntos de datos específicos a un dominio
- Anotaciones manuales son muy costosas

Entrenamiento mediante tareas de pretexto

- reconstrucción de la imagen de entrada
- predicción de píxeles en una secuencia de video
- ordenamiento temporal de cuadros de videos
- ordenamiento parches en imágenes estáticas

Anotaciones fáciles de obtener

Hipótesis: el modelo aprenderá características de alto nivel para poder realizar su tarea.

Automovimiento

Supervisar aprendizaje mediante información odométrica

¿Podemos predecir cuando dos fotos son la misma a pesar haber sufrido alguna transformación?

- Tarea de clasificación donde el objetivo es predecir la transformación

Automovimiento

¿Como sabemos si son buenas representaciones?

Automovimiento

¿Como sabemos si son buenas representaciones?

- ① la capacidad de poder realizar múltiples tareas visuales
- ② la habilidad de poder aprender nuevas tareas visuales basándose en pocos ejemplos.

Automovimiento

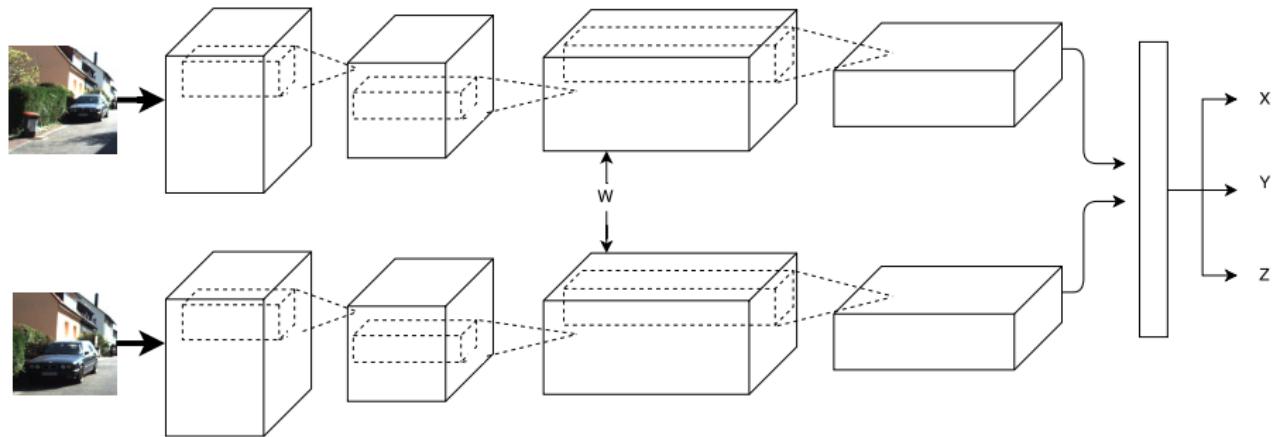
¿Como sabemos si son buenas representaciones?

- ① la capacidad de poder realizar múltiples tareas visuales
- ② la habilidad de poder aprender nuevas tareas visuales basándose en pocos ejemplos.

El problema de relacionar los estímulos visuales con el automovimiento puede ser establecido como el problema de predecir las transformaciones de la cámara en pares de imágenes a lo largo del movimiento de la misma

Automovimiento

El problema puede ser planteado con redes siamesas



Slow Feature Analysis

Técnica de aprendizaje no supervisado:

- *features* cambian poco en una ventana de tiempo pequeña

Slow Feature Analysis

Técnica de aprendizaje no supervisado:

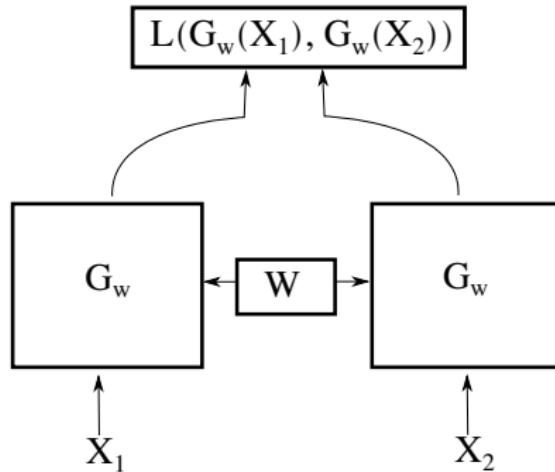
- *features* cambian poco en una ventana de tiempo pequeña

x_{t_1} y x_{t_2} representaciones de imágenes

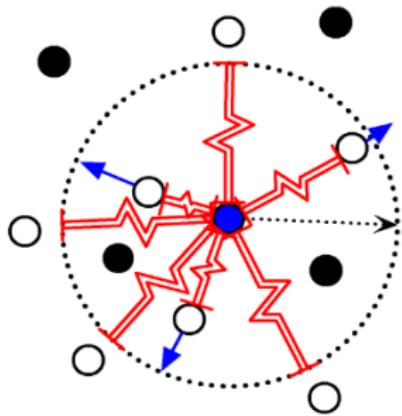
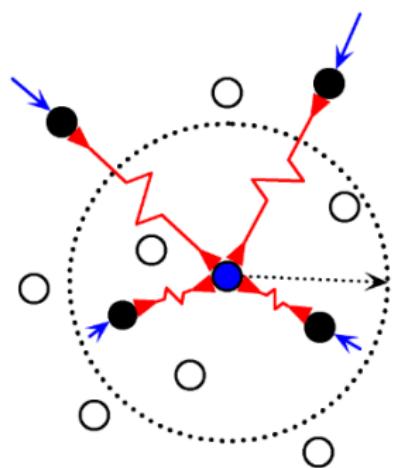
D una medida de similitud

$$L(x_{t_1}, x_{t_2}, W) = \begin{cases} D(x_{t_1}, x_{t_2}), & \text{si } |t_1 - t_2| \leq T \\ \max(0, m - D(x_{t_1}, x_{t_2})), & \text{si } |t_1 - t_2| > T \end{cases} \quad (7)$$

Slow Feature Analysis



Slow Feature Analysis



Información odométrica para entrenar redes siamesas

- Tarea de clasificación: predecir transformaciones en X, Y, Z
- Transformaciones en espacio continuo
- 3 clasificadores con categorías discretizadas

Experimentos - Diseño

- ① Particionado del conjunto de entrenamiento
- ② Hiperparámetros
- ③ Preentrenamiento inicial
- ④ Transferencia de aprendizaje
- ⑤ Chequeo de métricas

Experimentos - Prueba de concepto: MNIST

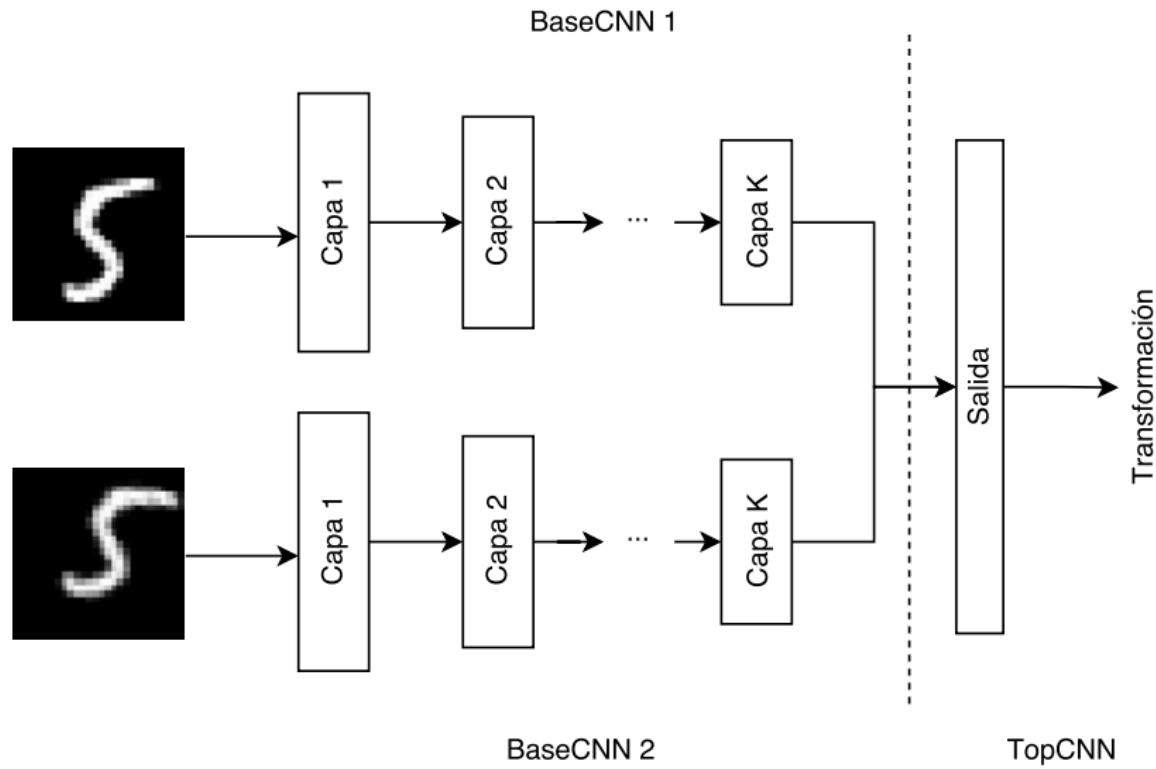
Conjunto de datos de dígitos manuscritos

Pares artificiales con traslaciones en X e Y, rotaciones en Z



- Traslaciones de [-3, 3] píxeles en X, Y
- Rotaciones de [-30, 30] grados en Z
- Para SFA, se consideran cercanos las imágenes en [-1,1], [-3, 3] grados

Experimentos - Prueba de concepto: MNIST



Experimentos - Prueba de concepto: MNIST

- Componentes siameses: C96-P-C256-P, Concatenación de representaciones: F1000-D-Op.
- 40K iteraciones, márgenes 10 y 100 para SFA
- 5 millones de pares de imágenes
- Partición en entrenamiento-validación
- Transferencia de aprendizaje: se congelaron las capas convolucionales.

Métricas

Accuracy, Función de costo durante entrenamiento

Experimentos - Prueba de concepto: MNIST

| Método | datos entrenamiento | | | |
|----------------|---------------------|-------------|-------------|-------------|
| | 100 | 300 | 1000 | 10000 |
| Desde cero | 0.42 | 0.70 | 0.82 | 0.97 |
| SFA($m=10$) | 0.52 | 0.71 | 0.77 | 0.82 |
| SFA($m=100$) | 0.58 | 0.73 | 0.80 | 0.88 |
| Automovimiento | 0.75 | 0.90 | 0.92 | 0.99 |

Experimentos - KITTI

11 secuencias que registran el movimiento de un automóvil en una ciudad



- Matrices de rotación Las transformaciones en X, Y y Z en base a las matrices de rotación provistas por los creadores del conjunto de datos.
- Para las rotaciones en el eje Y se calculó el ángulo correspondiente al cambio entre dos *frames*.
- Discretización de traslaciones y rotaciones en categorías

KITTI - Pares de entrenamiento



- Clasificación, 20 clases por eje
- Automovimiento: cuadros separados a lo sumo por 7 cuadros intermedios.
- SFA: 7 cuadros considerados cercanos
- Se extrajeron parches aleatorios de 227×227

KITTI - Entrenamiento

- Red: AlexNet
- Pre-entrenamiento 60K iteraciones
- Transferencia de aprendizaje: 10K iteraciones

KITTI - Entrenamiento

- Red: AlexNet
- Pre-entrenamiento 60K iteraciones
- Transferencia de aprendizaje: 10K iteraciones

Baseline

Se entrenó AlexNet con el conjunto de datos ILSVRC'12 desde cero utilizando 20 y 1000 imágenes por clase (ALEX-20 y ALEX-1000) Las redes siamesas fueron entrenadas con aproximadamente 20K pares de imágenes.

KITTI - Entrenamiento

- Red: AlexNet
- Pre-entrenamiento 60K iteraciones
- Transferencia de aprendizaje: 10K iteraciones

Baseline

Se entrenó AlexNet con el conjunto de datos ILSVRC'12 desde cero utilizando 20 y 1000 imágenes por clase (ALEX-20 y ALEX-1000)

Las redes siamesas fueron entrenadas con aproximadamente 20K pares de imágenes.

Experimentos - KITTI + SUN397

397 categorías de paisajes interiores y exteriores

3 particiones con datos de entrenamiento-validación



Experimentos - KITTI + SUN397

Preentrenamiento: automovimiento, SFA y pesos aleatorios.

Transferencia de aprendizaje con SUN397 con 5 y 20 elementos por categoría.

Calidad de las representaciones: *Softmax* para cada capa

Experimentos - KITTI + SUN397

Preentrenamiento: automovimiento, SFA y pesos aleatorios.

Transferencia de aprendizaje con SUN397 con 5 y 20 elementos por categoría.

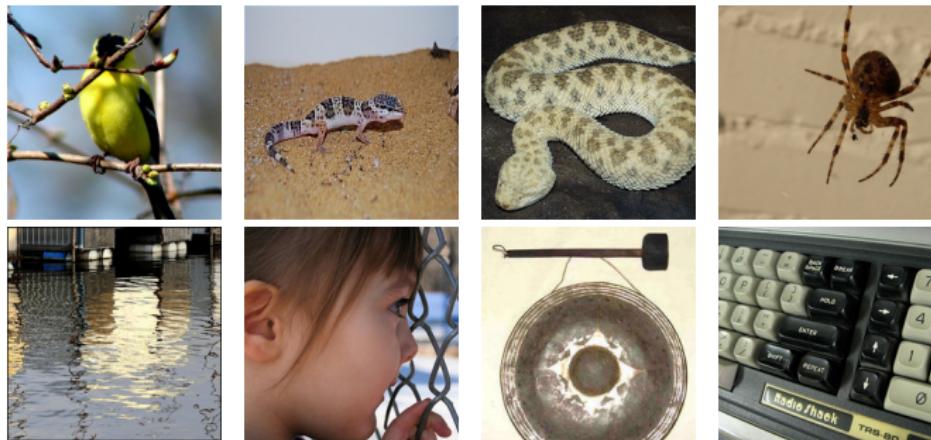
Calidad de las representaciones: *Softmax* para cada capa

Accuracy:

| Método | #preentr. | #finet. | L1 | L2 | L3 | L4 | L5 | #finet. | L1 | L2 | L3 | L4 | L5 |
|-----------|-----------|---------|-------------|-------------|-------------|-------------|-------------|---------|-------------|-------------|-------------|-------------|-------------|
| ALEX-1000 | 1M | 5 | 3.73 | 5.07 | 5.07 | 8.53 | 10.40 | 20 | 9.07 | 12.53 | 16.27 | 17.60 | 10.67 |
| ALEX-20 | 20K | 5 | 2.93 | 1.87 | 3.73 | 5.07 | 3.20 | 20 | 6.13 | 5.33 | 5.33 | 4.53 | 5.07 |
| KITTI-SFA | 20.7K | 5 | 2.13 | 3.20 | 2.40 | 1.60 | 1.87 | 20 | 4.53 | 3.73 | 2.13 | 2.40 | 2.93 |
| KITTI-AUT | 20.7K | 5 | 2.93 | 1.87 | 3.20 | 5.87 | 1.33 | 20 | 6.67 | 7.47 | 9.87 | 9.33 | 4.00 |

Experimentos - KITTI + ImageNet

Conjunto de datos ILSVRC'12, 1000 categorías



Se crearon conjuntos de datos con 1 5, 10, 20 y 1000 imágenes por categoría

Experimentos - KITTI + ImageNet

Preentrenamiento con automovimiento, SFA y pesos aleatorios.

Experimentos - KITTI + ImageNet

Preentrenamiento con automovimiento, SFA y pesos aleatorios.

Transferencia de aprendizaje con ImageNet.

Experimentos - KITTI + ImageNet

Preentrenamiento con automovimiento, SFA y pesos aleatorios.

Transferencia de aprendizaje con ImageNet.

Accuracy:

| Método | 1 | 5 | 10 | 20 | 1000 |
|-----------|-------------|-------------|-------------|-------------|-------------|
| KITTI-AUT | 0.49 | 1.27 | 2.14 | 4.13 | 20.8 |
| KITTI-SFA | 0.35 | 0.75 | 1.34 | 2.64 | 11.83 |
| ALEXNET | 0.45 | 0.95 | 1.91 | 3.69 | 18.35 |

Conclusiones

- *automovimiento* permite obtener resultados equiparables al estado del arte.
- Anotaciones de automovimiento fáciles de obtener.
- Bajo la misma cantidad de imágenes de entrenamiento, *automovimiento* equipara a técnicas supervisadas de clasificación.

Trabajo a Futuro

- Probar otras variantes: arquitecturas de redes, conjuntos de datos
- ¿Puede el *automovimiento* aplicarse a otros dominios?
- Evaluar la utilidad de otras tareas de pretexto

¿Preguntas?

Gracias!

