Redes Neuronales para Lenguaje Natural

2024

Grupo de Procesamiento de Lenguaje Natural
Instituto de Computación

Arquitecturas Secuenciales

Arquitecturas Secuenciales

Motivación:

Juan no vio la película que me gustó

Juan vio la película que no me gustó

¿Cómo los representamos en BOW?

¿Y con embeddings?

Arquitecturas Secuenciales

- El lenguaje está compuesto por palabras
- Secuencias de largo variable
- El orden de las palabras es muy relevante
- Pero los MLP tenían entrada de tamaño fijo…
- Sabemos representar una palabra (embeddings), y si pudiéramos ir presentándolas una a una en la red?

Redes Convolucionales

Redes Convolucionales

Convolutional Neural Network (CNN)

Tienen su origen en tareas de visión artificial

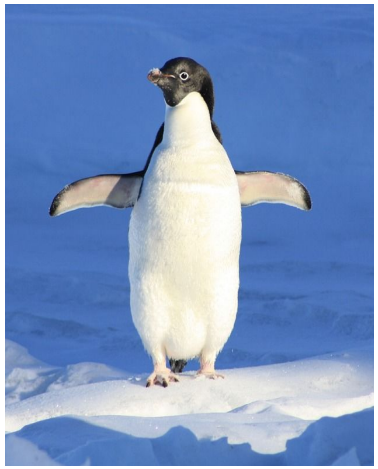
Ventajas directas en procesamiento de imágenes y videos

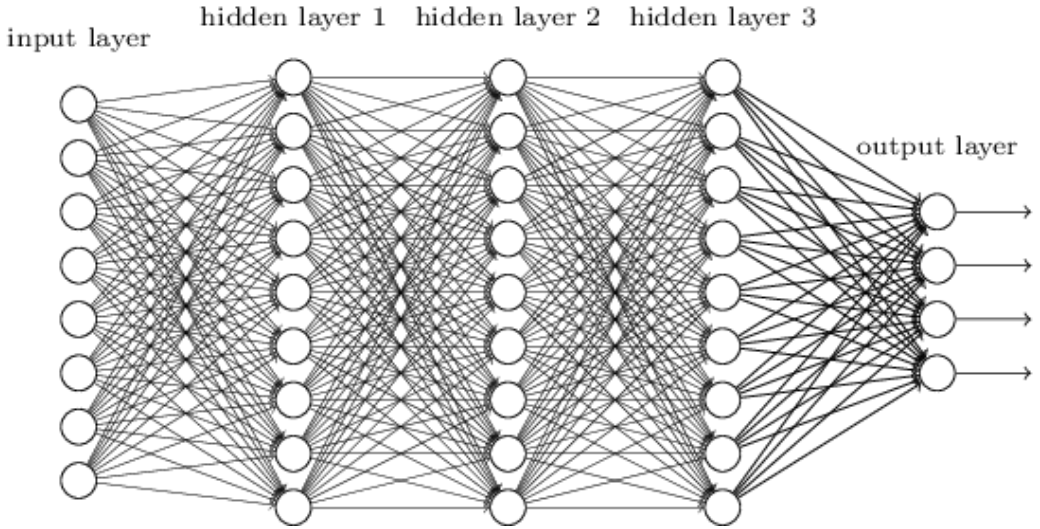
Es útil contar con invarianzas a distintas transformaciones

¿Cómo?

- A partir de los datos
- Que ocurra naturalmente por la estructura de la red

Limitaciones de MLPs en Imágenes





- Cantidad de parámetros a aprender (inicialización, overfitting)
- “Rigidez” frente al valor de cada pixel (aprender la invarianza de los datos)
- No se aprovecha la forma 2D de la imágen

Redes Convolucionales

Históricamente: **Neocognitrón (Fukushima, 1980)** se inspira en experimentos de estímulos y respuestas en las células del córtex visual de un gato **(Hubel y Wiesel, 1961)**.

Luego, se obtiene el estado del arte en clasificación de caracteres manuscritos con **LeNet (LeCun, 1998)**.

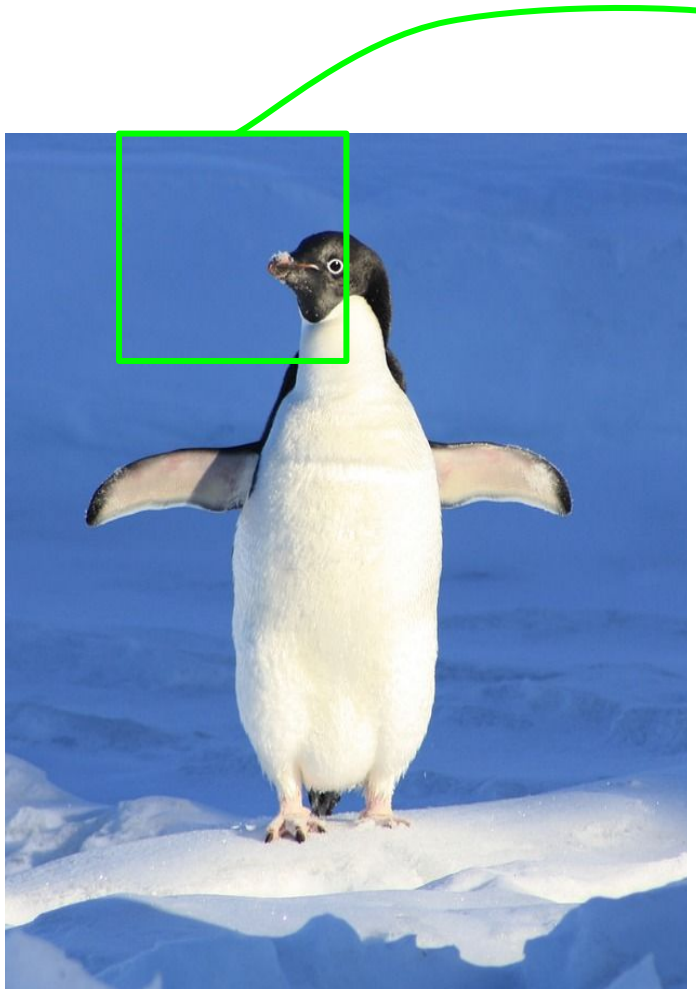
En 2012 **AlexNet (Krizhevsky, Sutskever & Hinton, 2012)** obtienen el estado del arte clasificando imágenes en 1000 clases

Redes Convolucionales

Las CNNs introducen dos componentes que consideran la forma bidimensional (o mayor) de la entrada

- Capas convolucionales
- Capas de pooling

Capa convolucional



Cada fragmento se pasa por una subred más pequeña (kernel o filtro) que devuelve un valor

Al pasar por todas las regiones de la imagen se va creando una nueva “imagen transformada”

Parámetros del kernel:

- Tamaño
- Stride
- Padding
- Profundidad / cantidad de salidas

Kernels

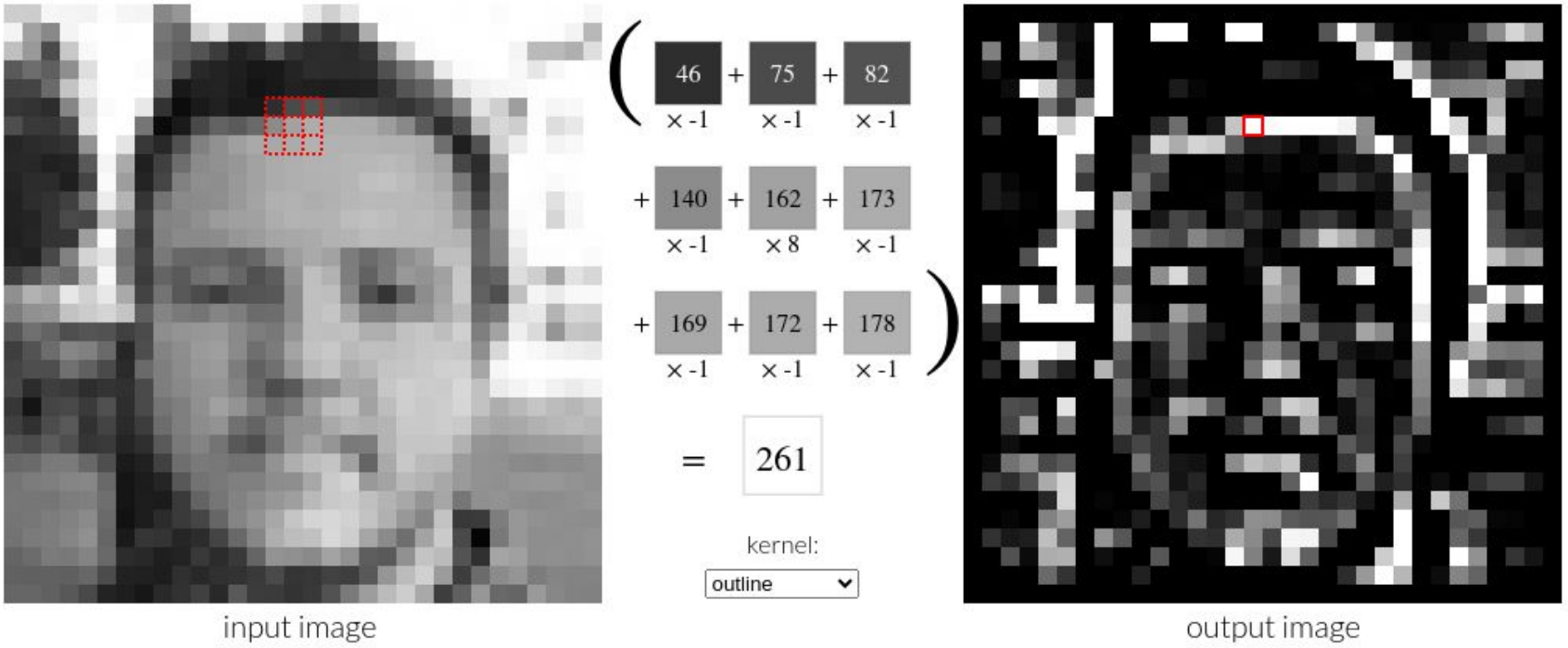
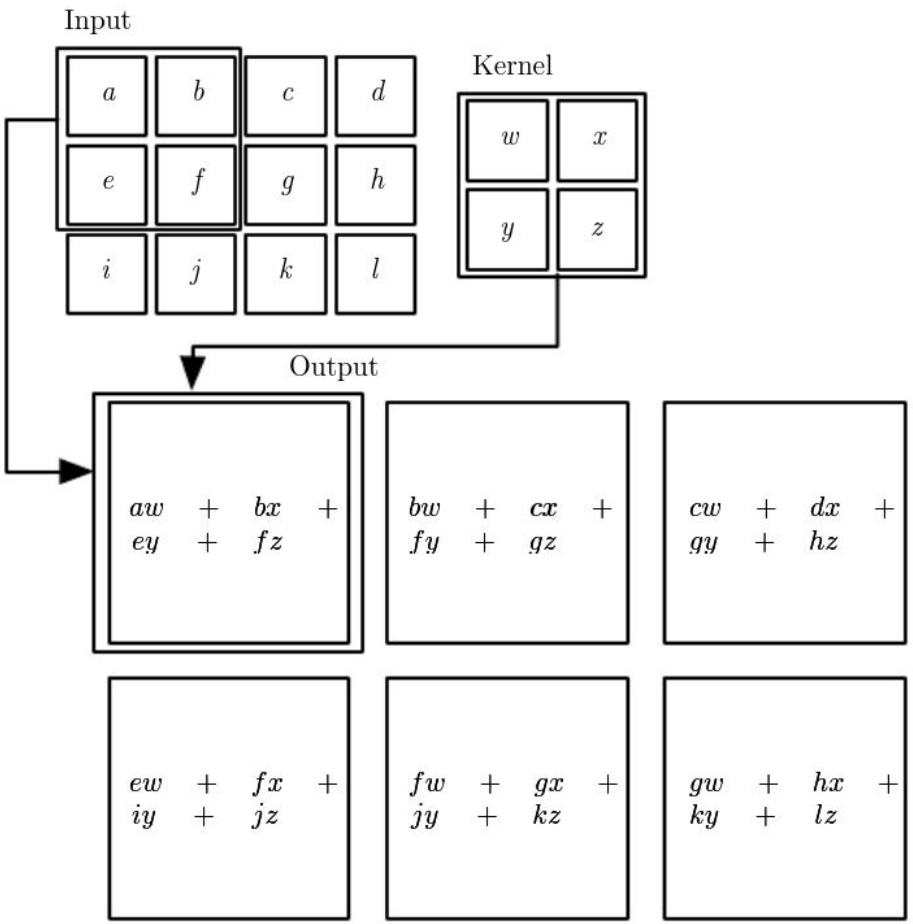


Image Kernels Explained Visually, Victor Powell
https://setosa.io/ev/image-kernels/

Kernels



1 kernel (o filtro)

- Tamaño 2x2
- Stride = 1

Notar que la convolución 2D es **equivariante** con respecto a traslaciones

Pensemos en “detectores” entrenables que procesan regiones de la imágen

Capa de Pooling

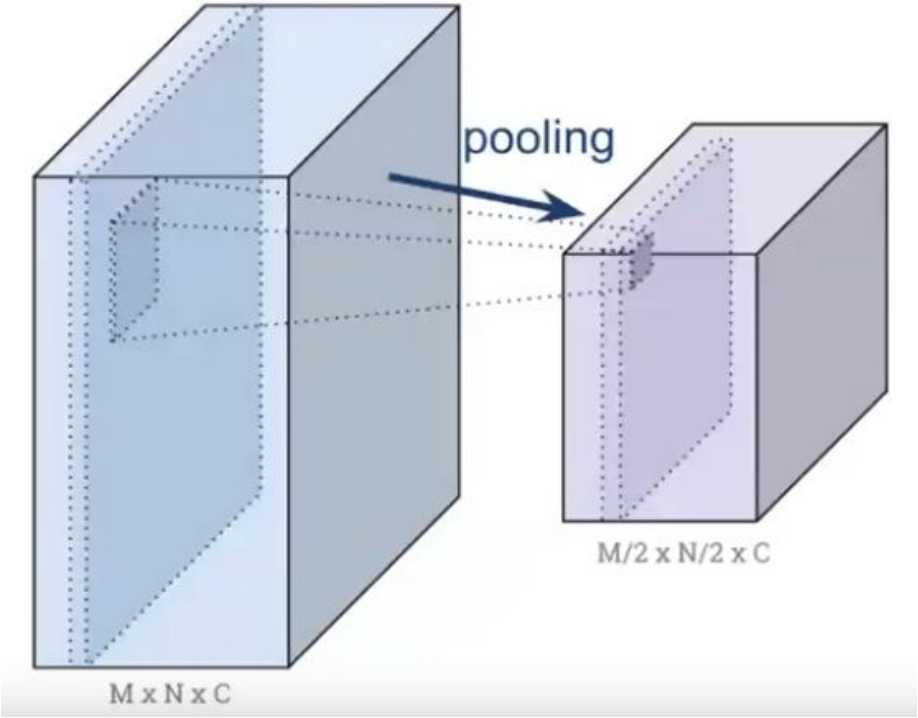
- Se aplican después de una capa convolucional

- Especie de “filtro sin parámetros” que aplica una operación a varios elementos (ej. máximo, promedio).
  - Intuitivamente, submuestrear la entrada “no cambia mucho las formas en la imágen” y reduce la cantidad de parámetros
  - Permite que el modelo aprenda **invariancias** locales

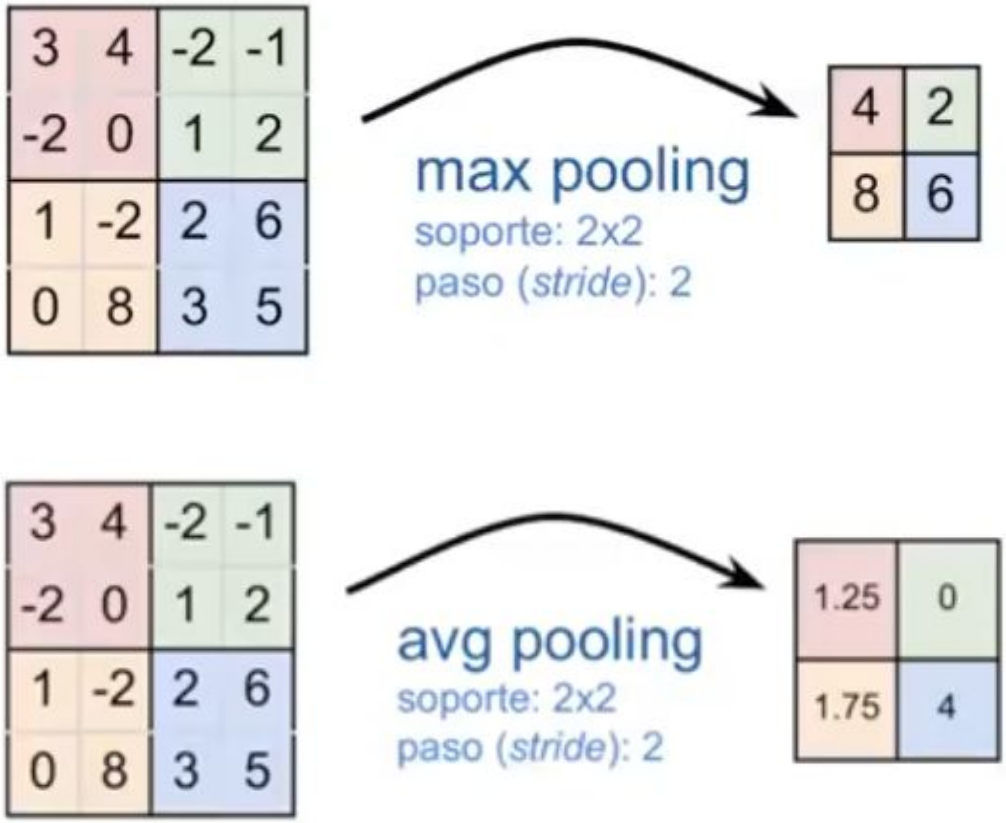
Capa de Pooling

Comprimir (sub-muestrear) la representación

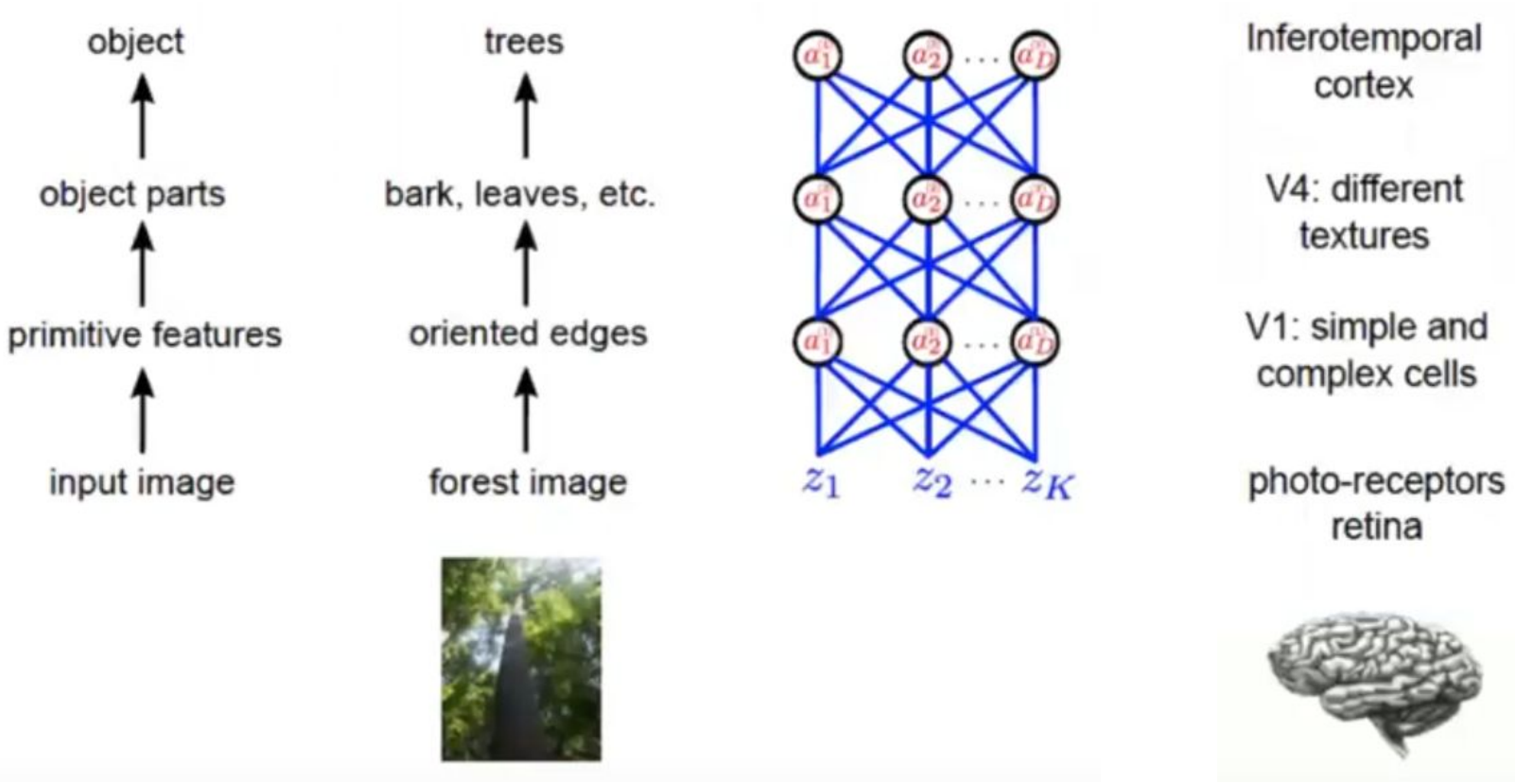
Se opera en cada canal por separado, por lo tanto se mantiene la cantidad de canales



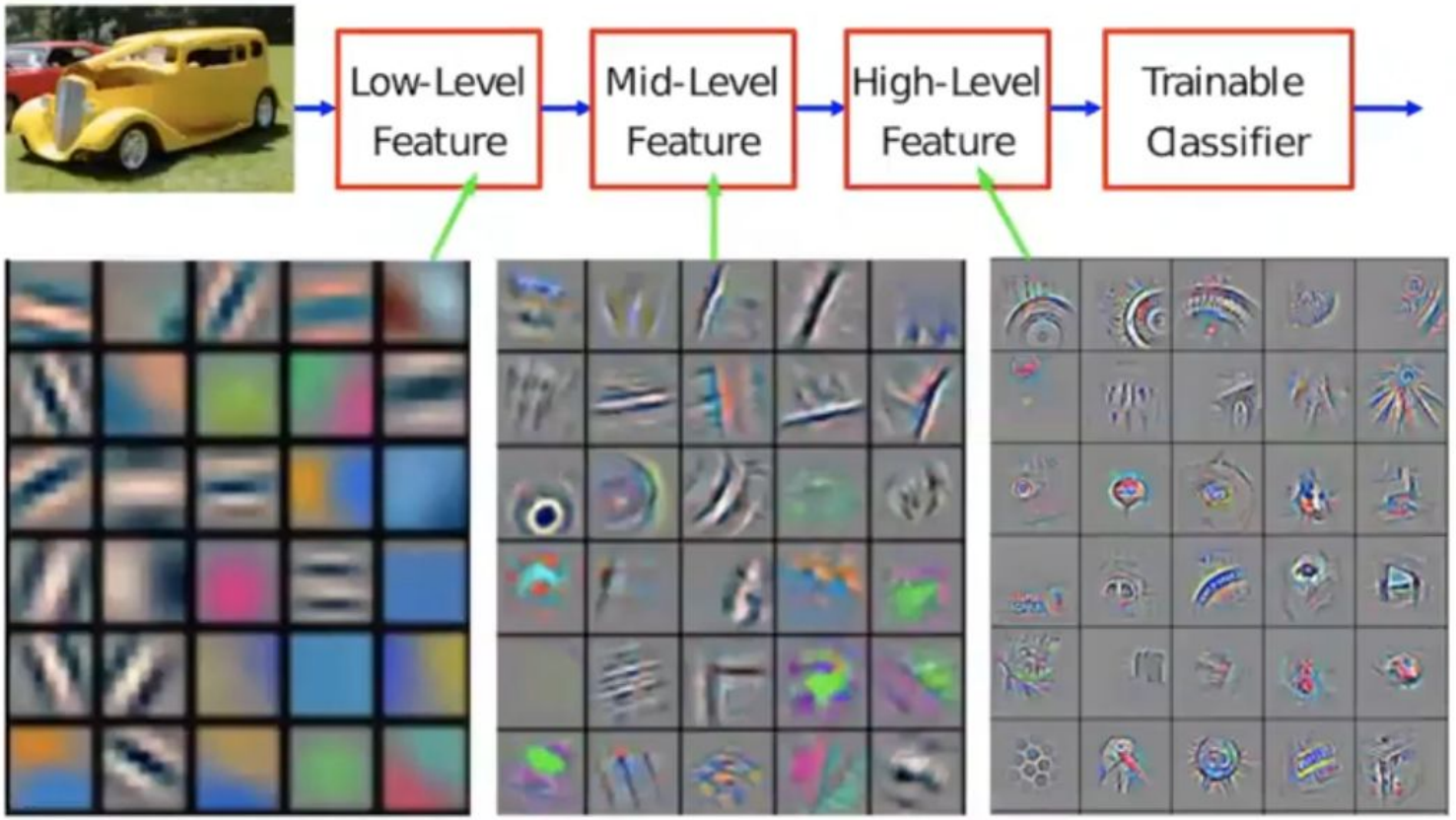
Capa de Pooling



Uso de varias capas convolucionales



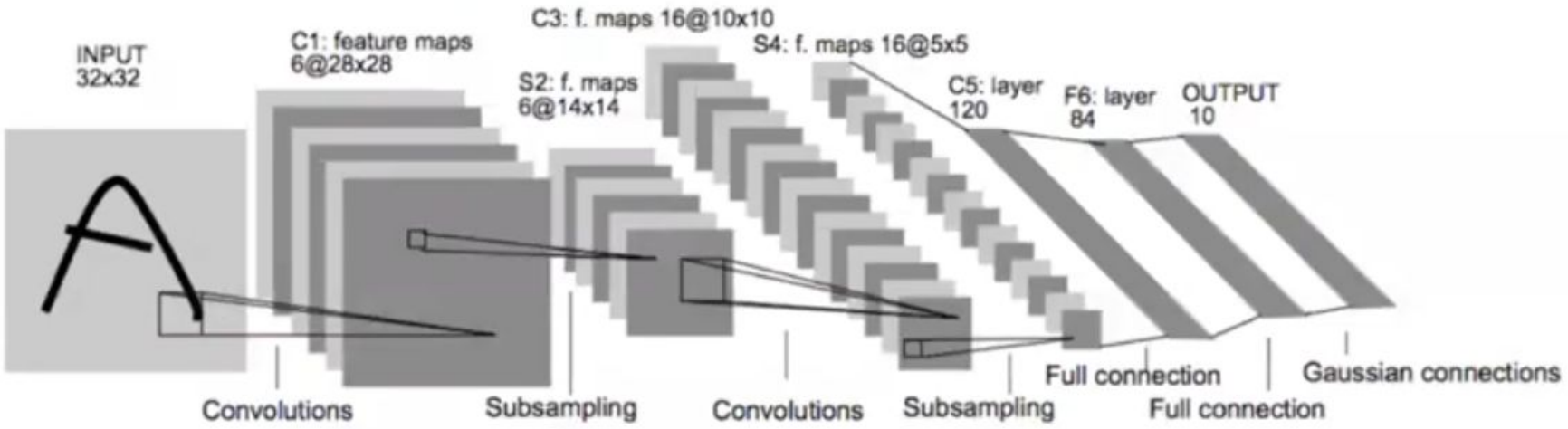
Uso de varias capas convolucionales



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

LeNet (1998)

Se utilizó para lectura de dígitos y caracteres manuscritos con resultados excelentes para la época.



Backpropagation applied to handwritten zip code recognition
LeCun et al., (1998)

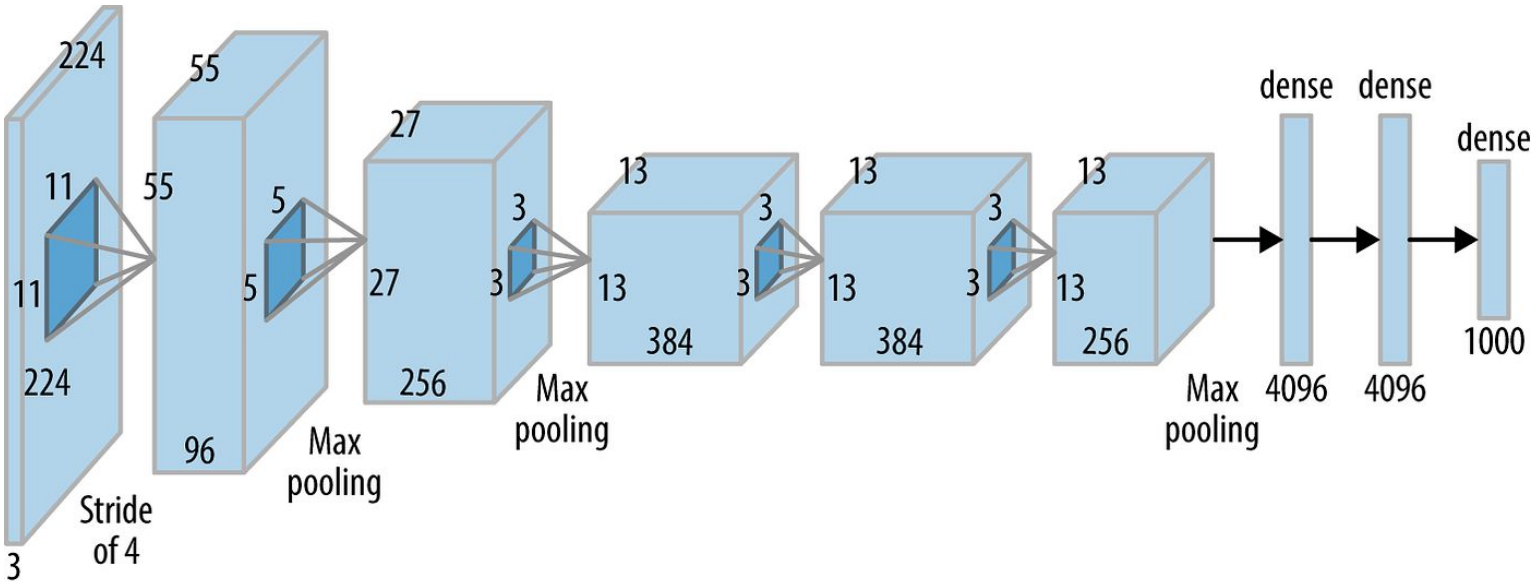
ImageNet (2009)

Idea iniciada en 2006 por Fei-Fei Li

Más de 1 millón de imágenes en 1000 clases diferentes

ImageNet: A large-scale hierarchical image database
Deng, Socher, Li, et al., (2009)

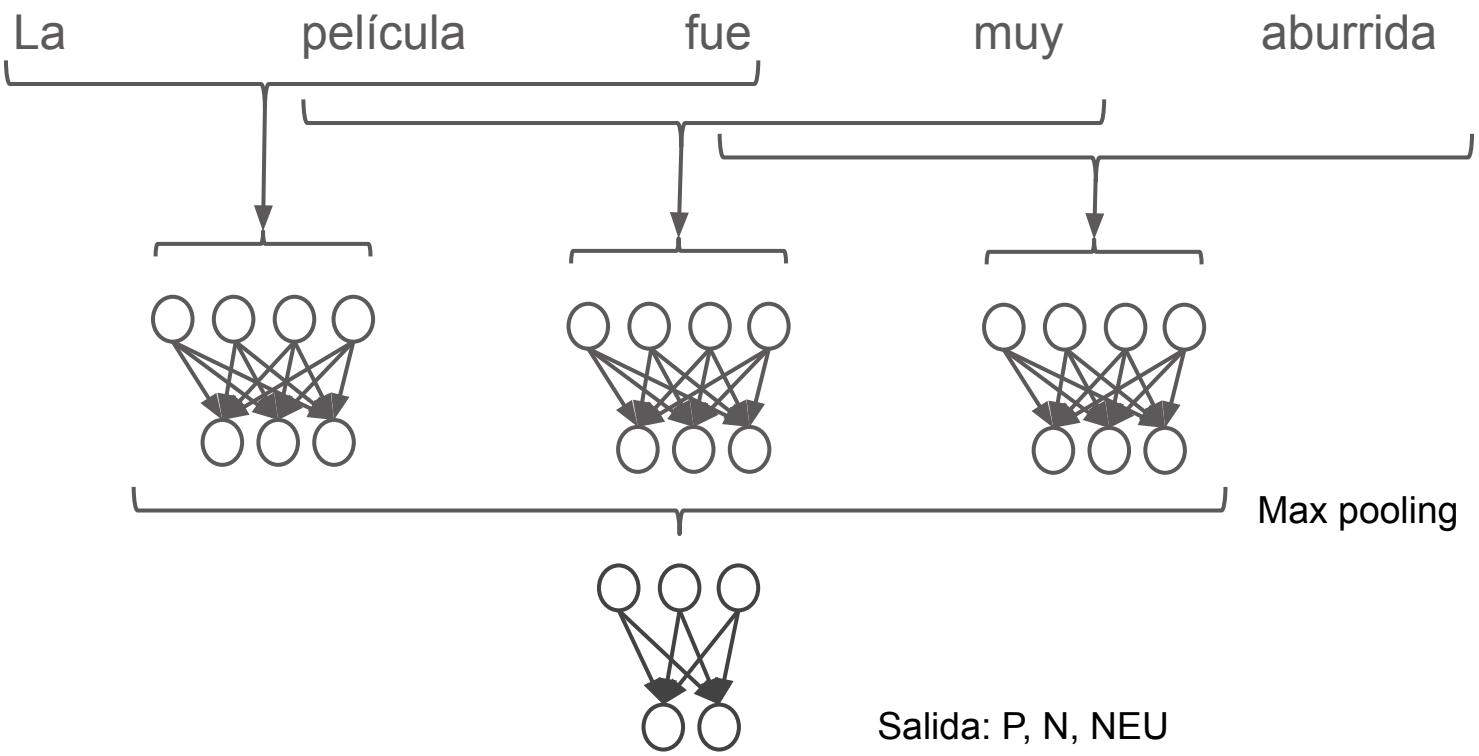
AlexNet (2012)



ImageNet Classification with Deep Convolutional Neural Networks
Krizhevsky, Sutskever & Hinton (2012)

CNNs para Lenguaje

Red Convolucional 1D



CNN para Lenguaje

“A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning”

Collobert & Weston (2008)

“Natural Language Processing (Almost) from Scratch”

Collobert, Weston et al. (2011)

Presentan una red convolucional adaptable a diferentes tareas de PLN

Entrenan todas las tareas a la vez, y eso obtiene mejoras en todas!

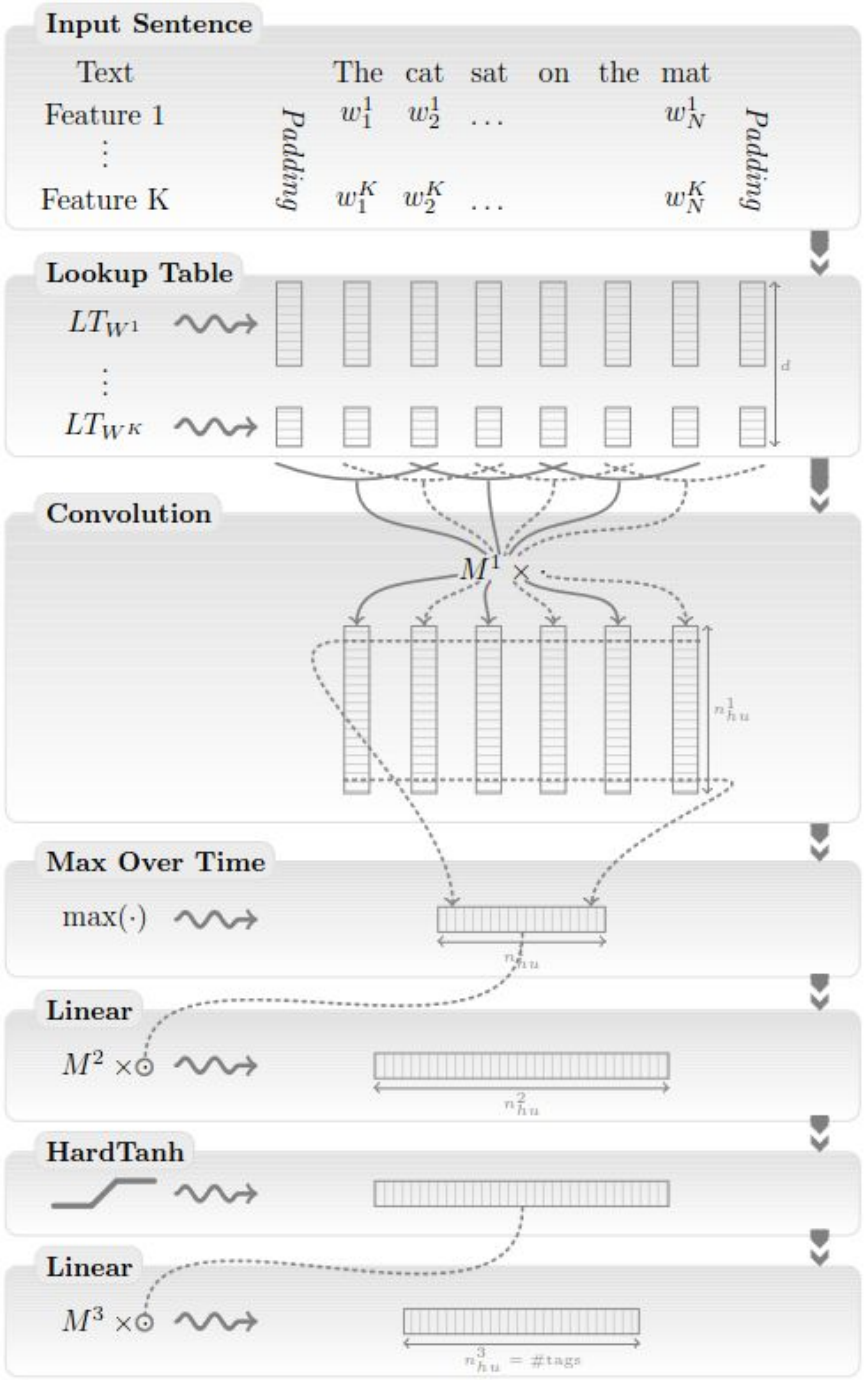
Varias tareas: Multi-task learning

Task	Benchmark	Data set	Training set (#tokens)	Test set (#tokens)	(#tags)
POS	Toutanova et al. (2003)	WSJ	sections 0–18 (912,344)	sections 22–24 (129,654)	(45)
Chunking	CoNLL 2000	WSJ	sections 15–18 (211,727)	section 20 (47,377)	(42) (IOBES)
NER	CoNLL 2003	Reuters	“eng.train” (203,621)	“eng.testb” (46,435)	(17) (IOBES)
SRL	CoNLL 2005	WSJ	sections 2–21 (950,028)	section 23 + 3 Brown sections (63,843)	(186) (IOBES)

Además: modelo de lenguaje (predecir la siguiente palabra) con Wikipedia

Todos los datos son en inglés

Todas las tareas se pueden modelar como etiquetado de secuencias (etiquetas BIO)



Natural Language Processing (Almost) from Scratch
Collobert, Weston et al. (2011)

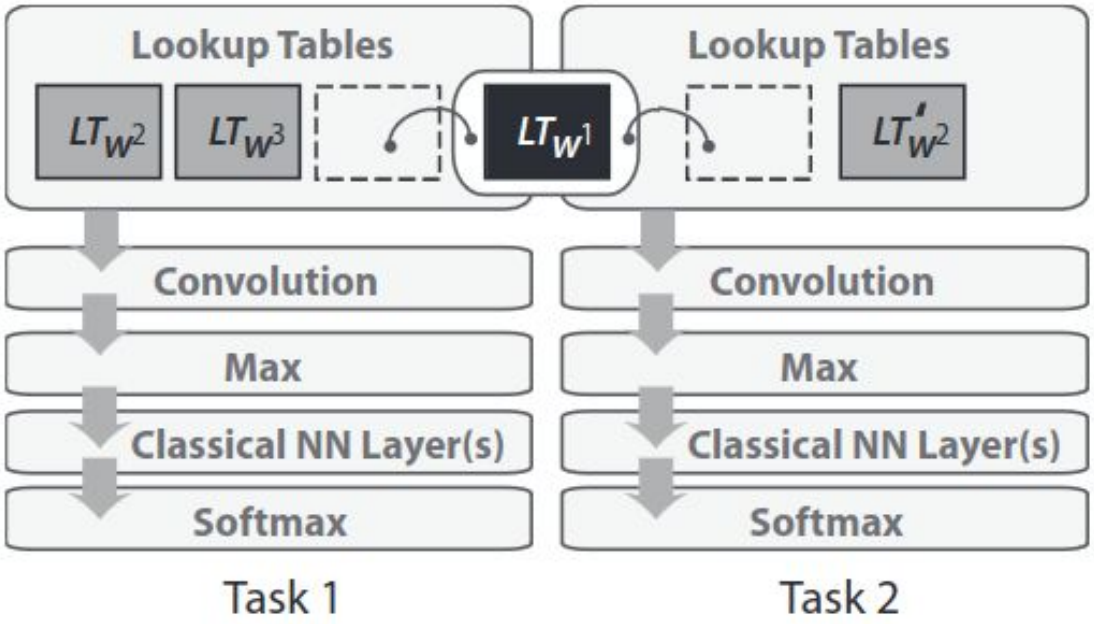
Varias tareas: Multi-task learning

¿Cómo lo adaptan a varias tareas?

En cada iteración sortean una tarea y un conjunto de ejemplos para ajustar

Las “lookup tables” (tablas de embeddings) se reutilizan en el entrenamiento

De la capa de convolución en adelante, se usa un stack diferente por tarea



Resultados

Obtuvieron mejoras para (casi) todas las tareas

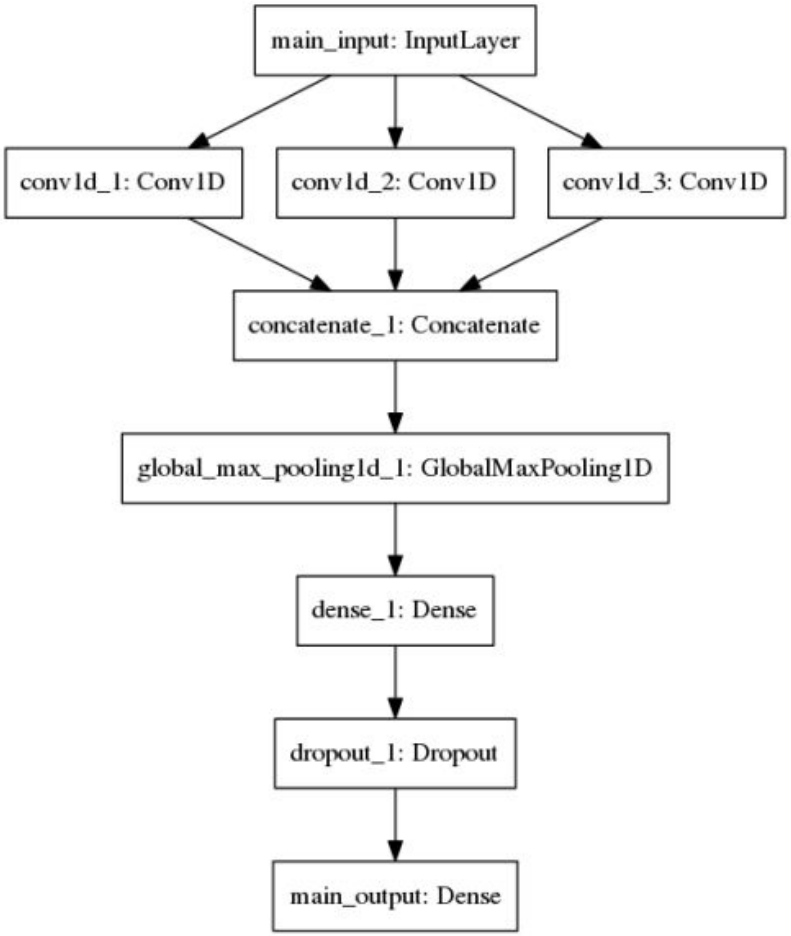
Usando una única arquitectura unificada y más “simple”

Task		Benchmark	SENNA
Part of Speech (POS)	(Accuracy)	97.24 %	97.29 %
Chunking (CHUNK)	(F1)	94.29 %	94.32 %
Named Entity Recognition (NER)	(F1)	89.31 %	89.59 %
Parse Tree level 0 (PT0)	(F1)	91.94 %	92.25 %
Semantic Role Labeling (SRL)	(F1)	77.92 %	75.49 %

Además, mediante la tarea de modelado de lenguaje se generaron buenos embeddings

FRANCE	JESUS	XBOX	REDDISH	SCRATCHED
454	1973	6909	11724	29869
SPAIN	CHRIST	PLAYSTATION	YELLOWISH	SMASHED
ITALY	GOD	DREAMCAST	GREENISH	RIPPED
RUSSIA	RESURRECTION	PSNUMBER	BROWNISH	BRUSHED
POLAND	PRAYER	SNES	BLUISH	HURLED
ENGLAND	YAHWEH	WII	CREAMY	GRABBED
DENMARK	JOSEPHUS	NES	WHITISH	TOSSED
GERMANY	MOSES	NINTENDO	BLACKISH	SQUEEZED
PORTUGAL	SIN	GAMECUBE	SILVERY	BLASTED
SWEDEN	HEAVEN	PSP	GREYISH	TANGLED
AUSTRIA	SALVATION	AMIGA	PALER	SLASHED

CNN para análisis de sentimiento



Clasif.	M-F_1	Acierto
svm	49.9	61.7
cnn4	50.2	64.1
svm_cnn	**50.9**	**64.7**

Corpus de desarrollo

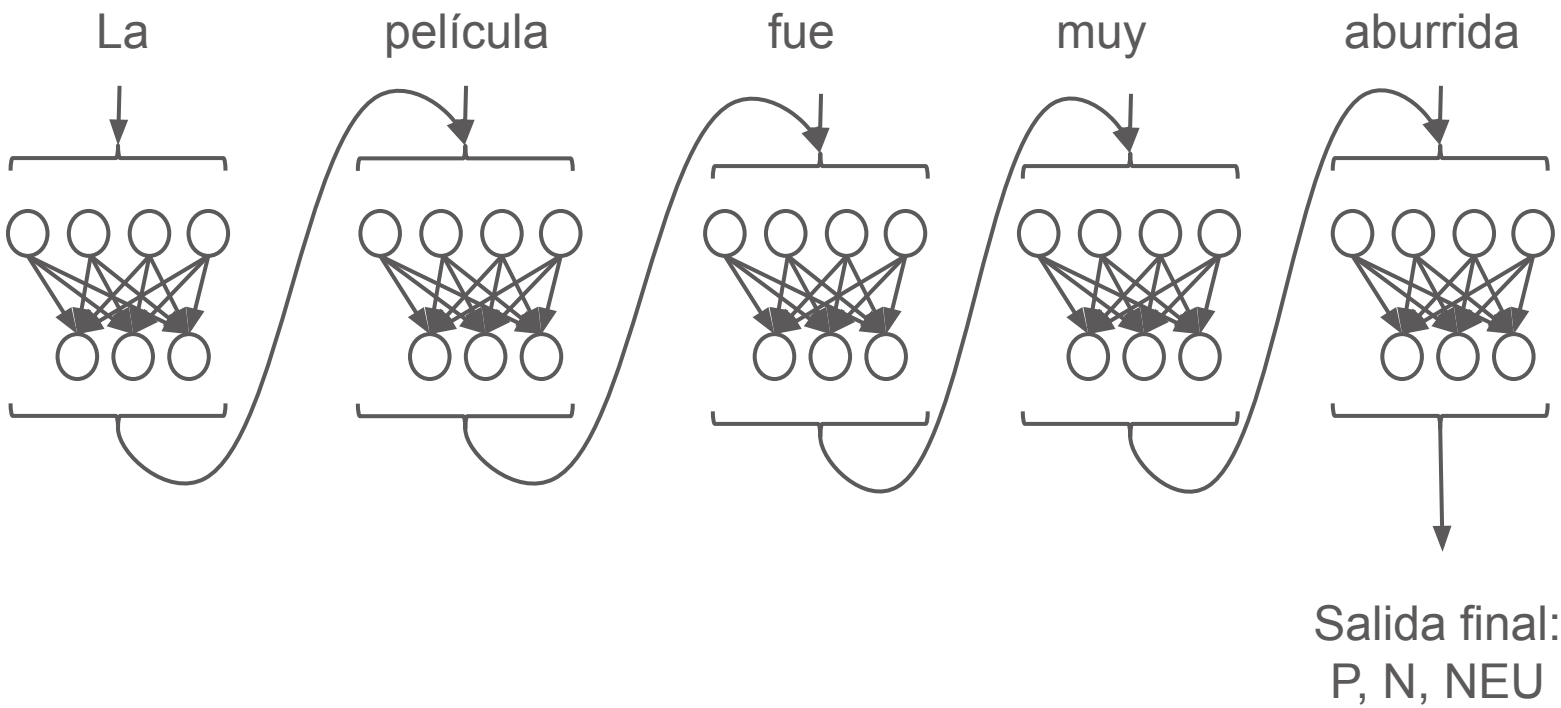
Corpus	Clasif.	M-F_1	Acierto
InterTASS	svm	45.7	58.3
InterTASS	cnn4	43.7	57.9
InterTASS	svm_cnn	**47.1**	**59.6**
Gral. TASS	svm	53.3	65.6
Gral. TASS	cnn4	53.1	66.5
Gral. TASS	svm_cnn	**54.6**	**67.4**
Gral. TASS 1k	svm	**56.2**	70.0
Gral. TASS 1k	cnn4	55.7	69.4
Gral. TASS 1k	svm_cnn	53.9	**71.1**

Corpus de test

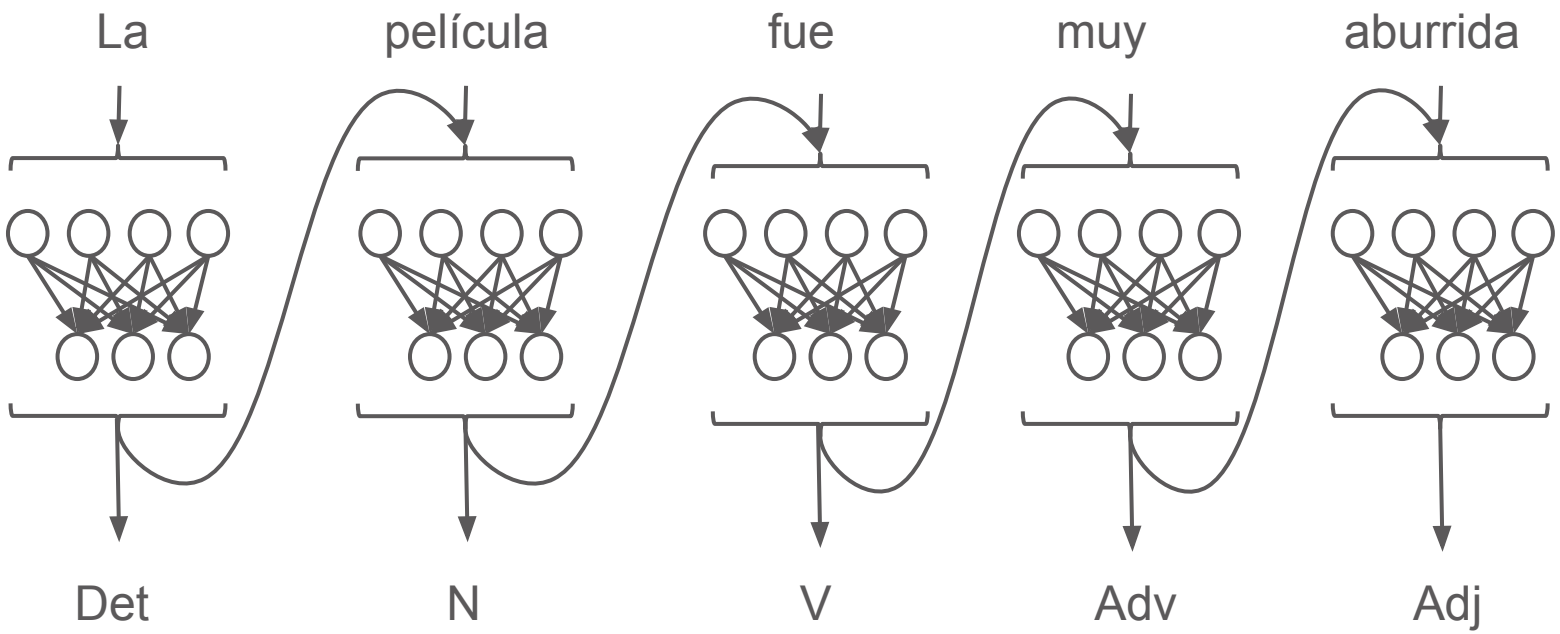
RETUYT in TASS 2017: Sentiment Analysis for Spanish Tweets using SVM and CNN
Rosá, Chiruzzo, Etcheverry & Castro (2017)

Redes Neuronales Recurrentes

Red Recurrente



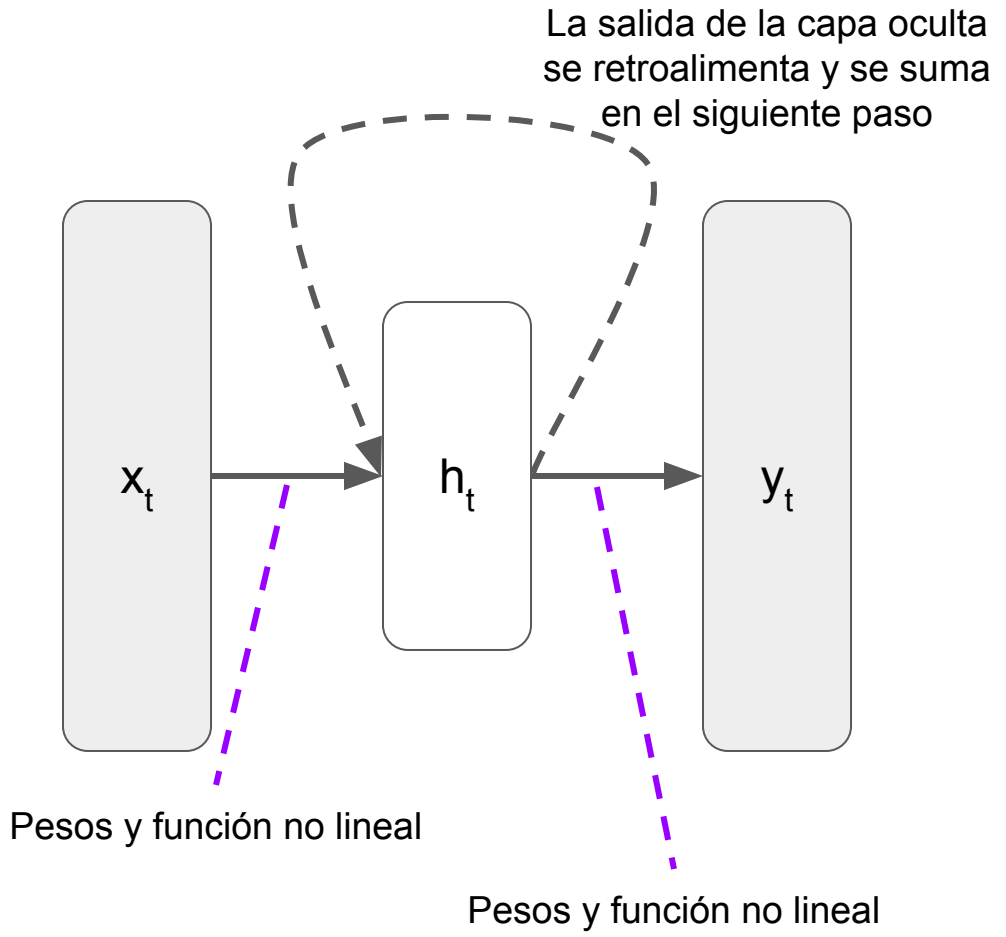
Red Recurrente - Salidas por palabra



Redes Recurrentes

- Recurrent Neural Network (RNN)
- Redes que contienen algún ciclo en sus conexiones
- El valor de alguna de las unidades está influido por valores anteriores de la entrada

Red Recurrente Simple (Red de Elman)



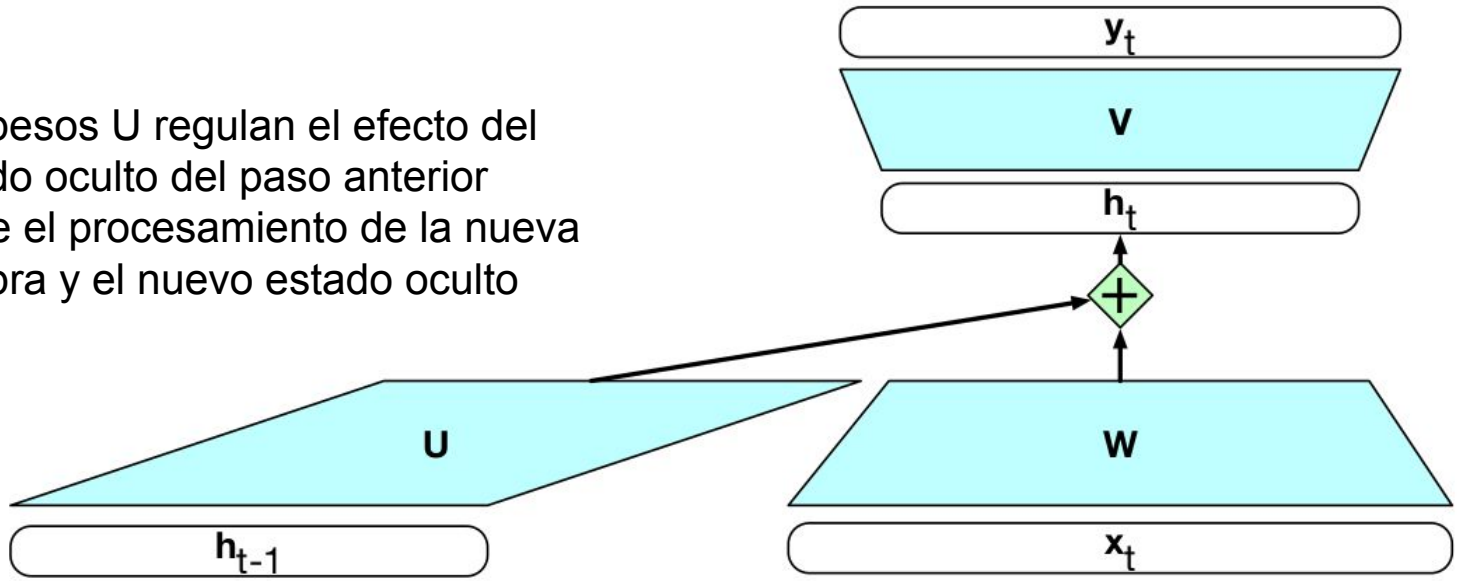
Finding Structure in Time
Elman (1990)

Red Recurrente Simple

- La capa oculta con conexión recurrente actúa como una especie de memoria
- Teóricamente podríamos presentar una secuencia de cualquier largo y podría recordarla toda
- Información del inicio de la oración puede tener influencia al final
  - Veremos que en la práctica esto es muy difícil

Red Recurrente Simple

Los pesos U regulan el efecto del estado oculto del paso anterior sobre el procesamiento de la nueva palabra y el nuevo estado oculto



$$h_t = g(Uh_{t-1} + Wx_t)$$

$$y_t = f(Vh_t)$$

U → dim_h x dim_h

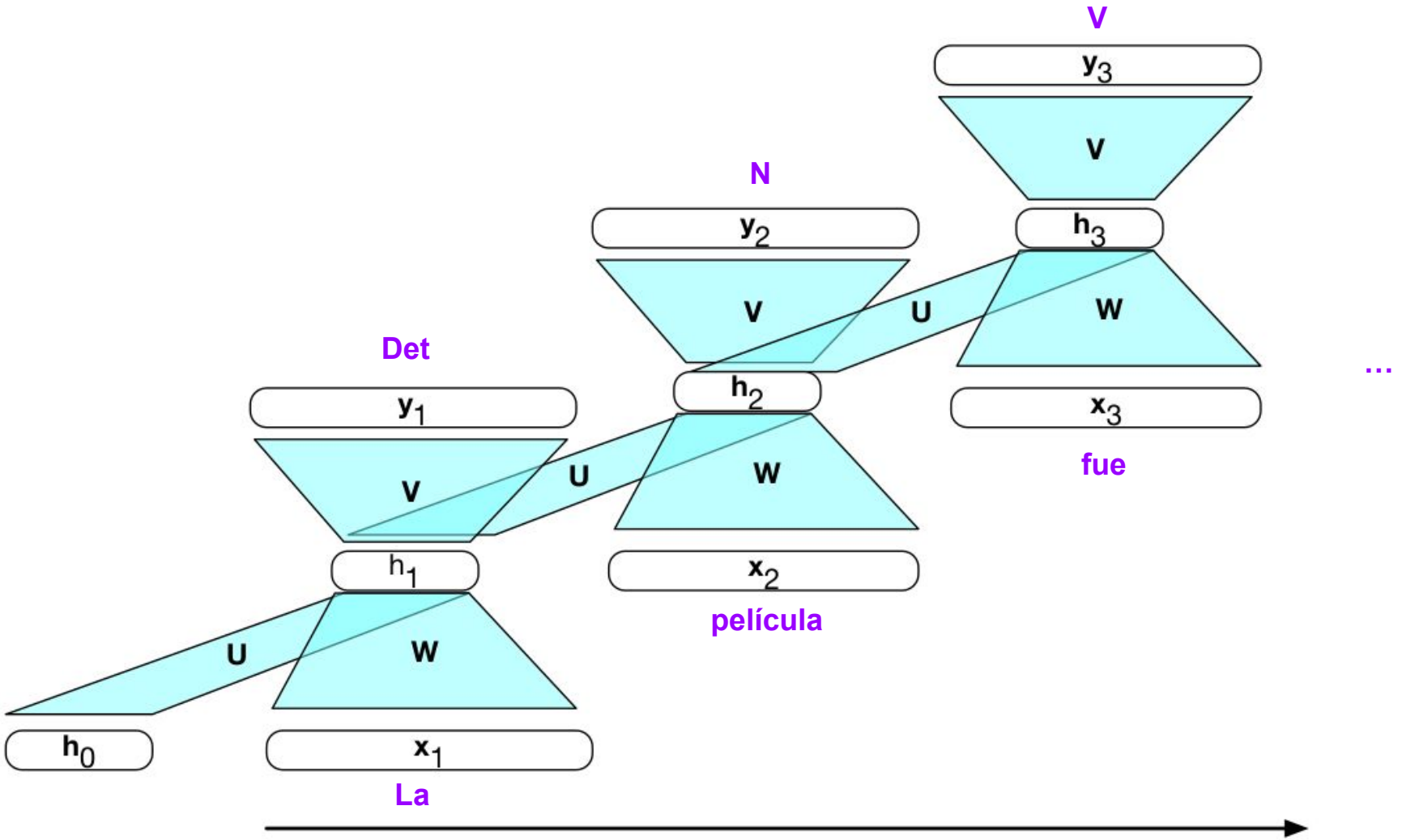
W → dim_h x dim_{in}

V → dim_{out} x dim_h

Por ejemplo:

$$y_t = \text{softmax}(Vh_t)$$

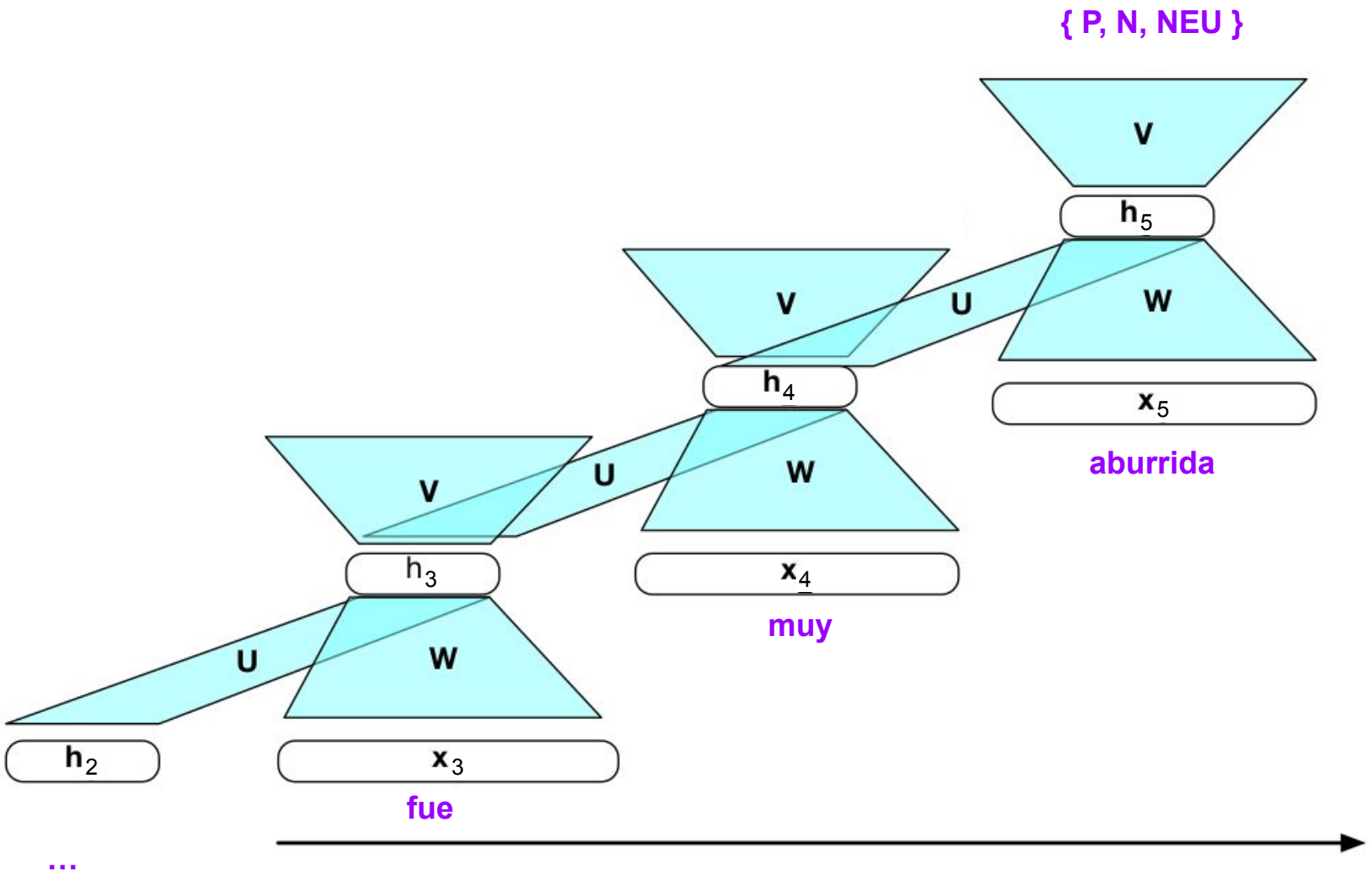
Desarrollo en el Tiempo



Entrenamiento

- Para cada ejemplo de entrenamiento, se aplica palabra a palabra desarrollando la red
- Cada una de las salidas tiene su propio *loss*
- El *loss* se va acumulando
- Backpropagation “en el tiempo”
- Los valores de h_{ti} influyen en todos los t_j, $j > i$

Desarrollo en el Tiempo



Entrenamiento

- En este caso, solo me interesa la salida final
- Todavía voy mostrando la entrada palabra a palabra y desarrollando la red
- El *loss* se calcula solo al final
- Qué tanto influyen las primeras palabras de la secuencia en el cálculo final?
  - Desvanecimiento de gradiente *(vanishing gradient)*
  - Ocurre por la naturaleza multiplicativa de las derivadas

Entrenamiento

- Presentando cada ejemplo por separado, el entrenamiento sería muy lento
- Pero las secuencias pueden tener largos diferentes
- Cómo crear un batch con secuencias de largos distintos?
  - Meter todos los ejemplos en un solo tensor
  - Padding de ceros al final
  - Que el tensor sea lo más pequeño posible para acomodar todo el batch

Tipos de problemas

1 palabra (o un par) → 1 categoría

frío y *caliente* son sinónimos?
antónimos?

MLP

n palabras → 1 categoría

este tweet tiene sentimiento positivo?
este tweet es un chiste?
este mail es spam?

MLP, CNN, RNN

n palabras → 0..n categorías

de qué temas habla este texto?
qué emociones presenta este tweet?

MLP, CNN, RNN

n palabras → n categorías

POS-tagging, NER, chunking,
parsing, SRL

CNN, RNN

n palabras → m palabras

traducción automática
respuestas a preguntas
resúmenes automáticos
chatbots…

Encoder-Decoder (RNN, Transformer)

Bibliografía

CNN:

Clase DLVIS2020: https://www.youtube.com/watch?v=esVvlYQSZog

Curso cs231n the stanford: https://cs231n.github.io/

Libro Goodfellow et al., 2016 : https://www.deeplearningbook.org/contents/convnets.html

Lecture 5 Smaller Network: CNN: https://slideplayer.com/slide/14085926/

Papers…

RNN:

Capítulo 8 del Jurafsky: https://web.stanford.edu/~jurafsky/slp3/8.pdf