# Redes Neuronales para Lenguaje Natural

2023

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*

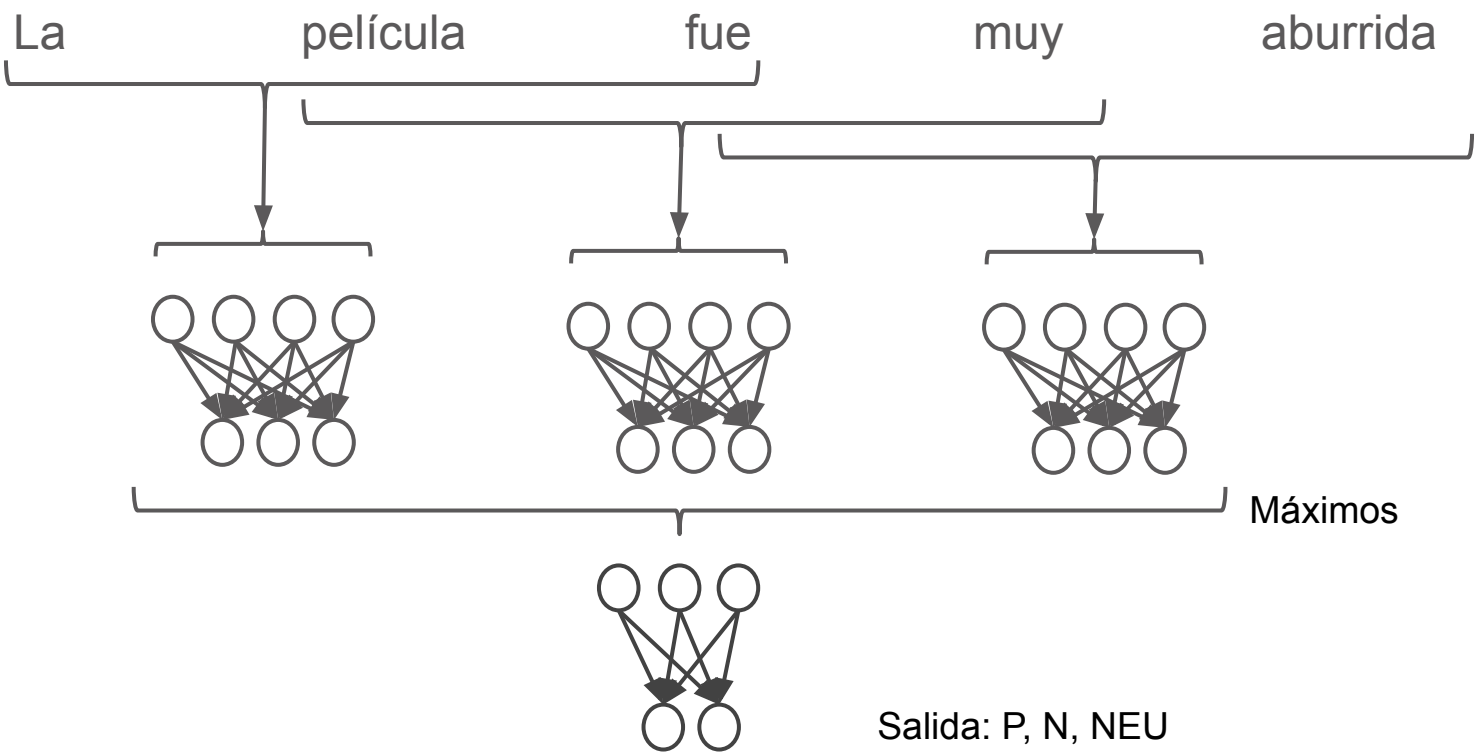
¿Cómo los representamos en BOW?

¿Y con embeddings?

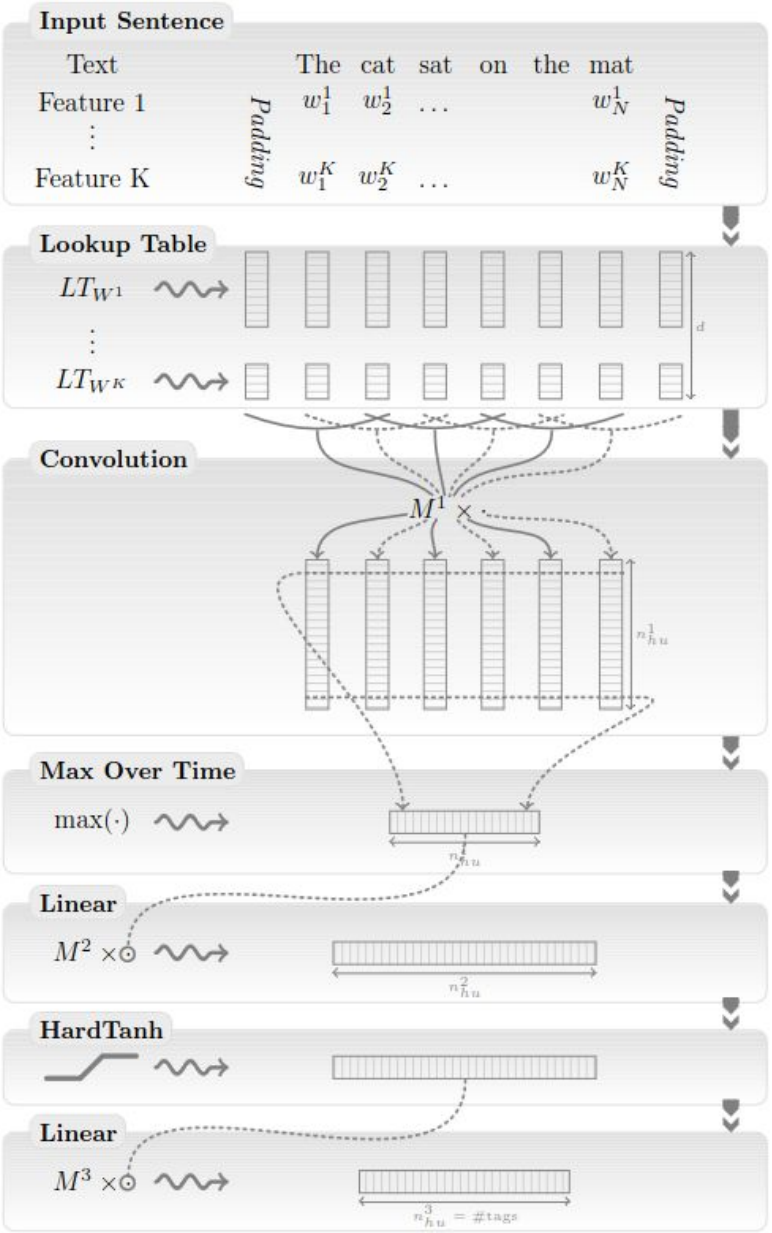
# Arquitecturas Secuenciales

- 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?

# Red Convolutiva



# CNN



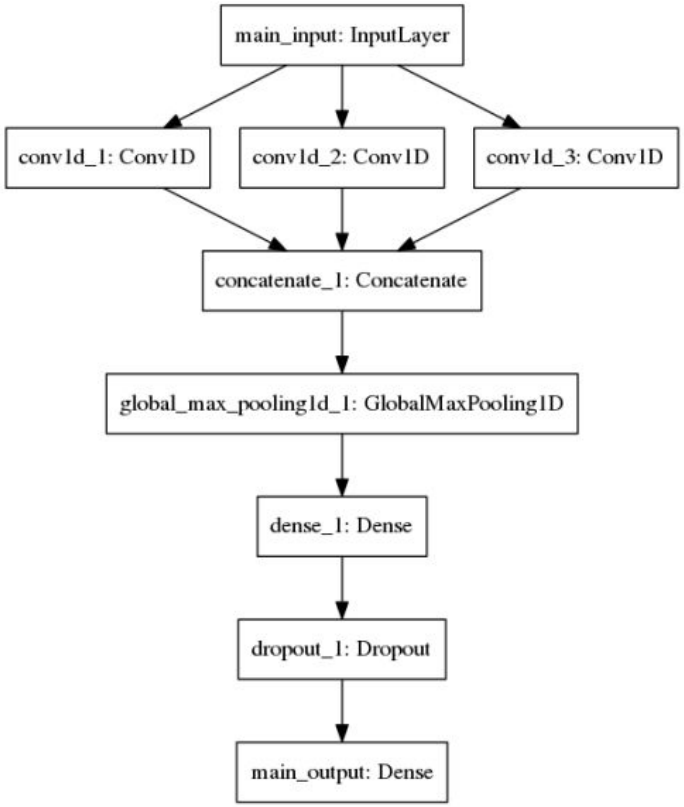

Natural Language Processing (Almost) from Scratch
Collobert, Weston et al. (2011)

# CNN

| 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) |

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

# 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

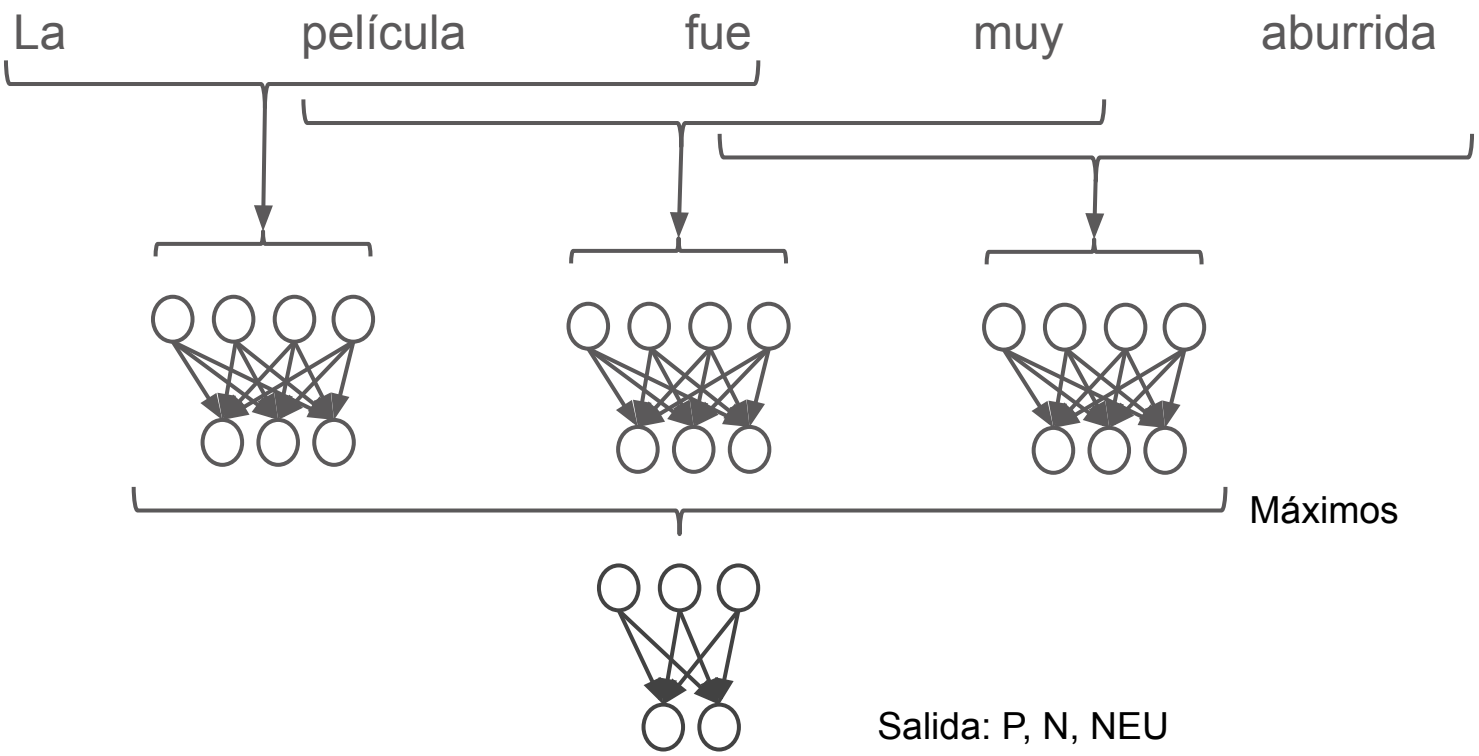
| Corpus | Clasif. | M-$F_1$ | Acierto |
|---|---|---|---|
| InterTASS | svm | 45.7 | 58.3 |
| InterTASS | cnn4 | 43.7 | 57.9 |
| InterTASS | svm_cnn | **47.1** | **59.6** |
| Gral. TASS | svm | 53.3 | 65.6 |
| Gral. TASS | cnn4 | 53.1 | 66.5 |
| Gral. TASS | svm_cnn | **54.6** | **67.4** |
| Gral. TASS 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

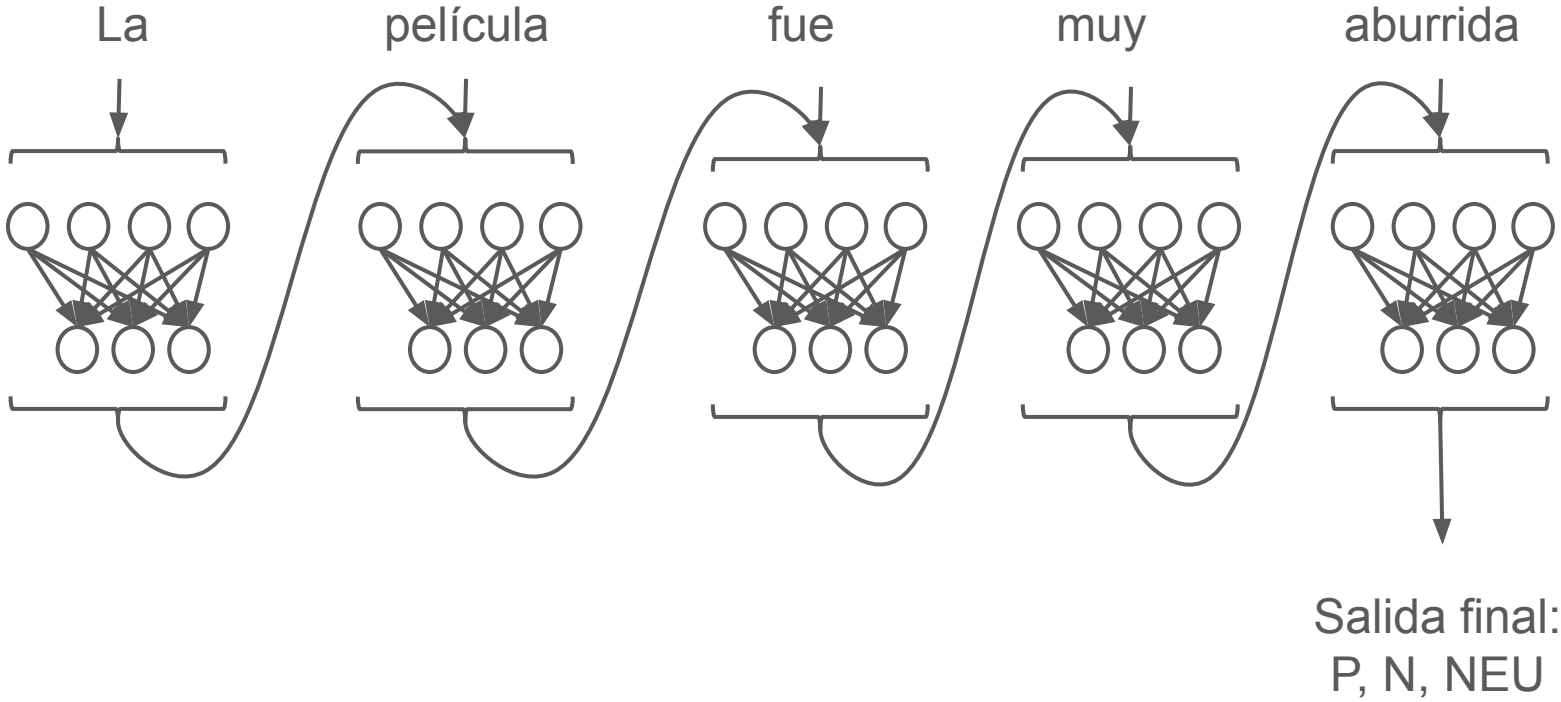
RETUYT in TASS 2017: Sentiment Analysis for Spanish Tweets using SVM and CNN
Rosá, Chiruzzo, Etcheverry & Castro (2011)

# Redes Neuronales Recurrentes

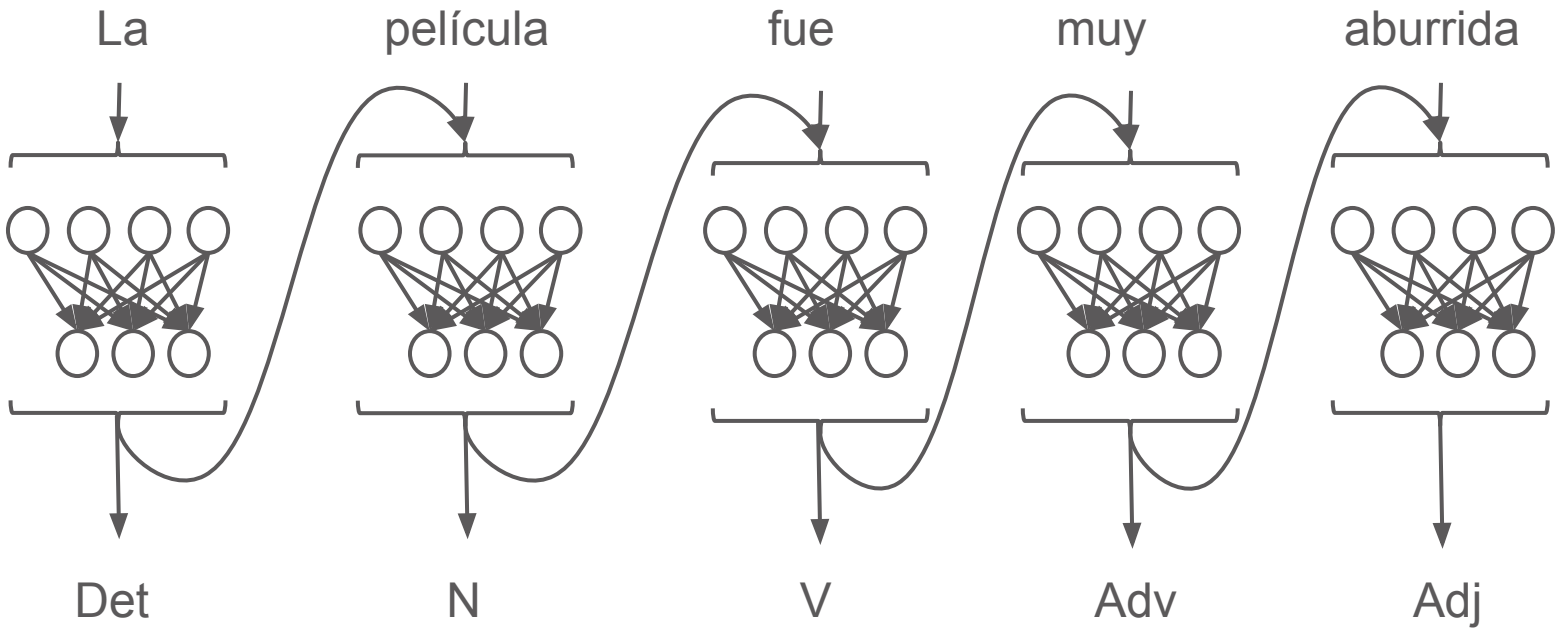
# Red Convolutiva



# Red Recurrente



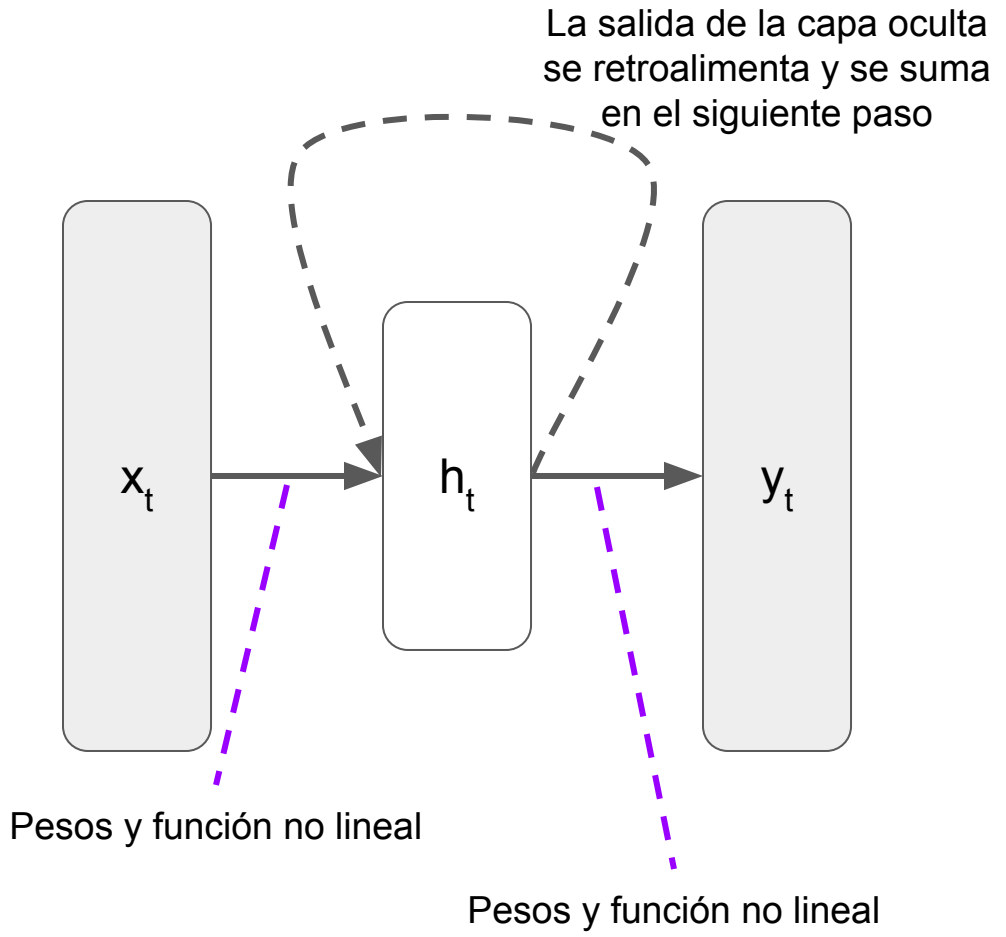
# Red Recurrente - Salidas por palabra



# 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)



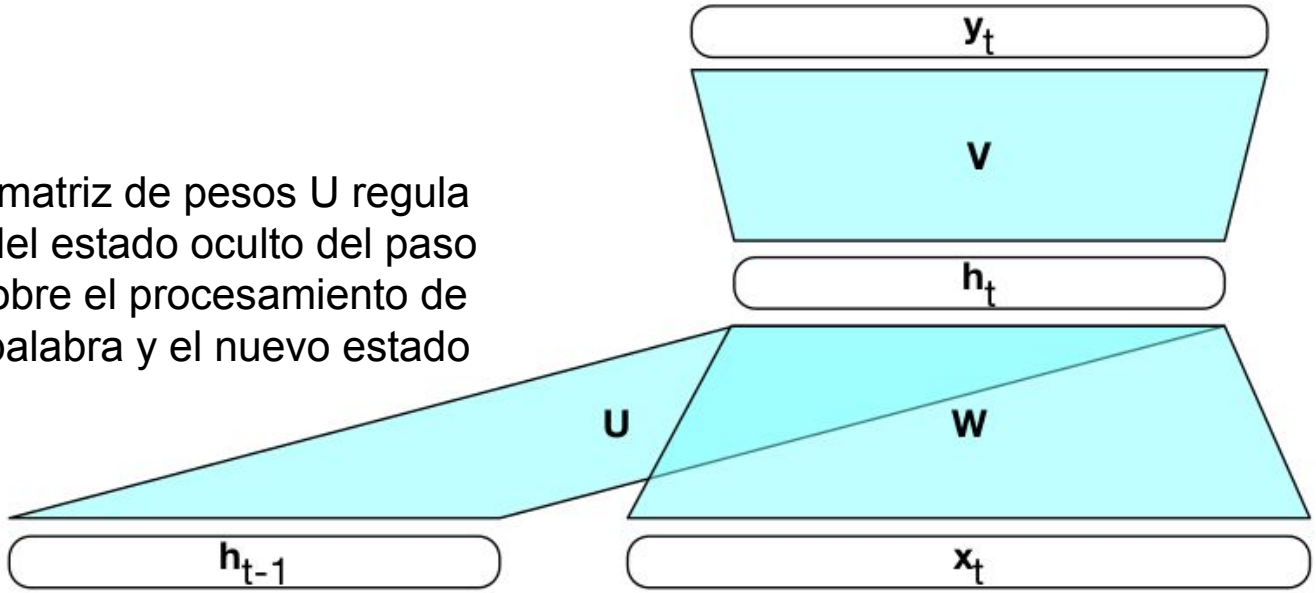

Finding Structure in Time
Elman (1990)

# 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

La nueva matriz de pesos U regula el efecto del estado oculto del paso anterior sobre el procesamiento de la nueva palabra y el nuevo estado oculto




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

$$y_t = f(Vh_t)$$

Por ejemplo:

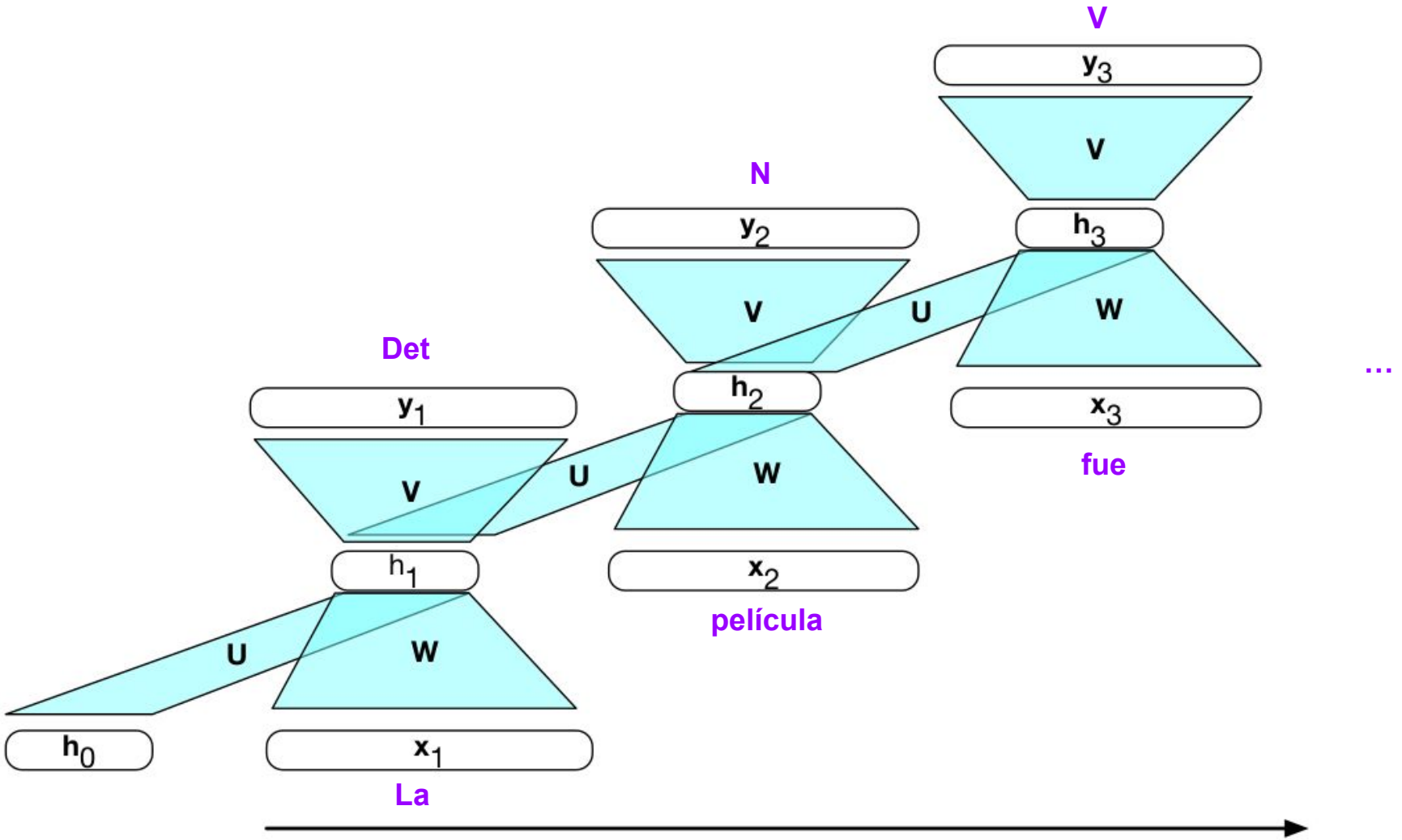
$$y_t = \text{softmax}(Vh_t)$$

$U \rightarrow dim_h \times dim_h$

$W \rightarrow dim_h \times dim_{in}$

$V \rightarrow dim_{out} \times dim_h$

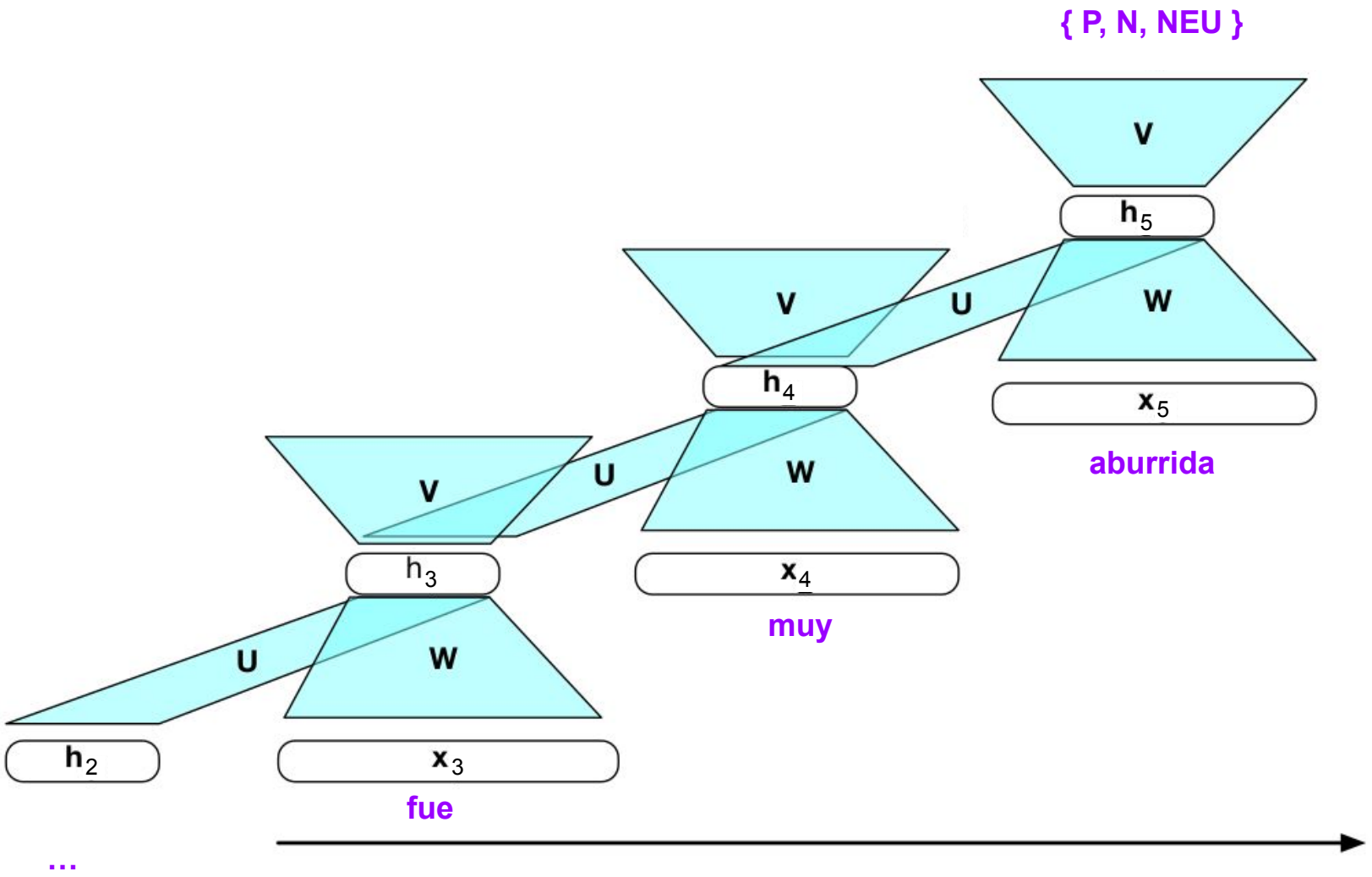
# Desarrollo en el Tiempo



# 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

# RNN como Modelos de Lenguaje

# Modelos de Lenguaje

*Debido a las copiosas* P=0.8 *lluvias* *de las últimas horas …*

P=0.09 *nevadas* *y avalanchas …*

P=0.0000001 *árbol*

- Predecir la probabilidad de una secuencia
- Predecir la siguiente palabra dado un prefijo

*P(lluvias | debido a las copiosas)*

$$P(w_{1:k}) = \prod_{i=1}^{k} P(w_i|w_{<i})$$

*P(<s>debido a las copiosas lluvias</s>) = P(<s>) P(debido|<s>)*
*P(a | <s> debido) P(las | <s> debido a) … P(</s> | <s>debido a las copiosas lluvias)*

# Modelos de Lenguaje

*P(<s>debido a las copiosas lluvias</s>) = P(<s>) P(debido|<s>)*
*P(a | <s> debido) P(las | <s> debido a) … P(</s> | <s>debido a las copiosas lluvias)*

Históricamente se resuelve mediante conteo de N-gramas

$$P(w_i|w_{<i}) \approx P(w_i|w_{i-N+1:i-1})$$
$$= P(w_i|w_{i-N+1}w_{i-N+2}\ldots w_{i-1})$$

el valor *N* de N-grama

El ejemplo anterior con tri-gramas:

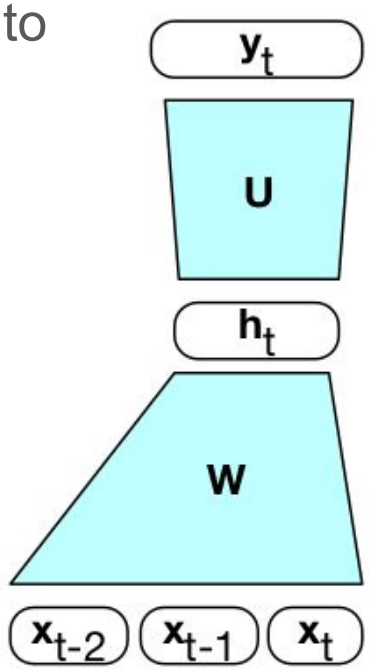
*P(<s>debido a las copiosas lluvias</s>) = P(<s>) P(debido|<s>)*
*P(a | <s> debido) P(las | debido a) P(copiosas | a las)*
*P(lluvias | las copiosas) P(</s> | copiosas lluvias)*

Problemas?

# Redes Neuronales como Modelo de Lenguaje

Podemos sustituir los N-gramas por una red feedforward

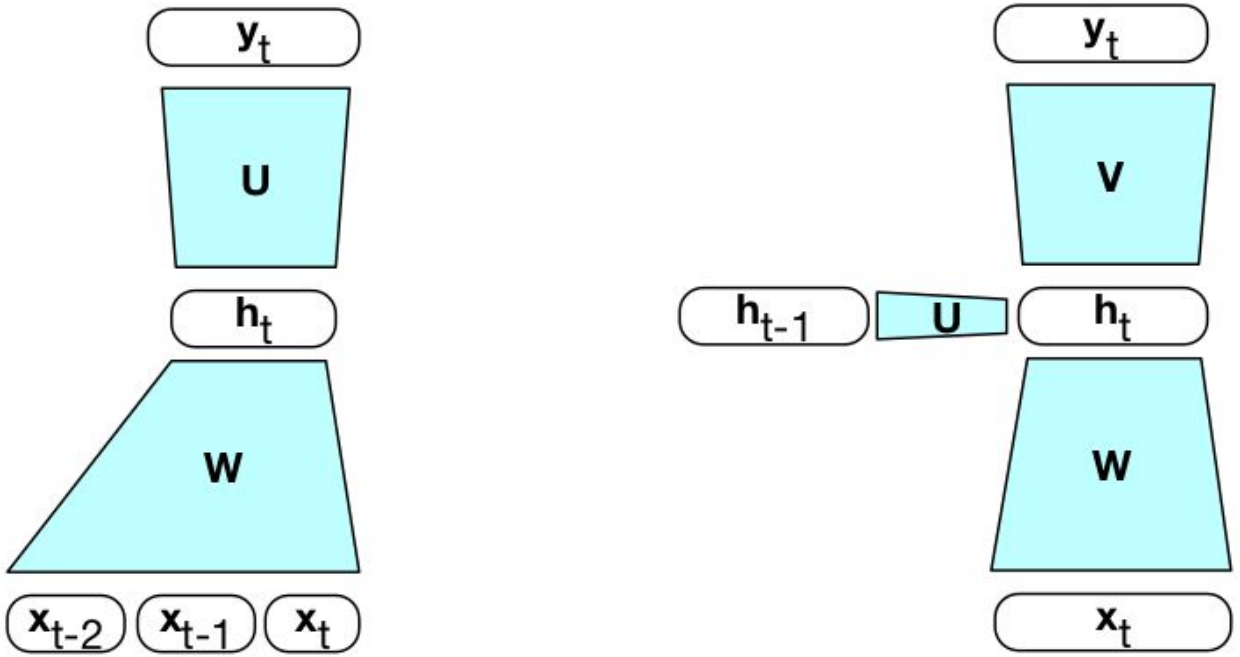
Mejora la generalización a ejemplos de N-gramas no vistos durante entrenamiento




Pero sigue el problema de que las palabras del comienzo pierden relevancia

# Redes Neuronales como Modelo de Lenguaje

Alternativa: Usar una RNN, de esta forma todo el contexto anterior se puede ir manteniendo en el estado oculto




Recurrent Neural Network based Language Model
Mikolov et al. (2010)

# RNN como Modelo de Lenguaje

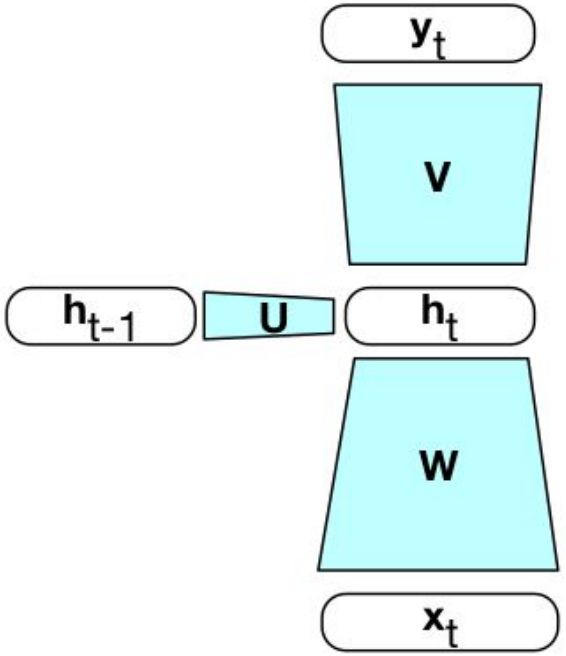
Tomamos la tabla de embeddings E

Matrices de pesos U, V y W

$$e_t = Ex_t$$
$$h_t = g(Uh_{t-1} + We_t)$$
$$y_t = \text{softmax}(Vh_t)$$

$y_t$ es un vector del tamaño del vocabulario

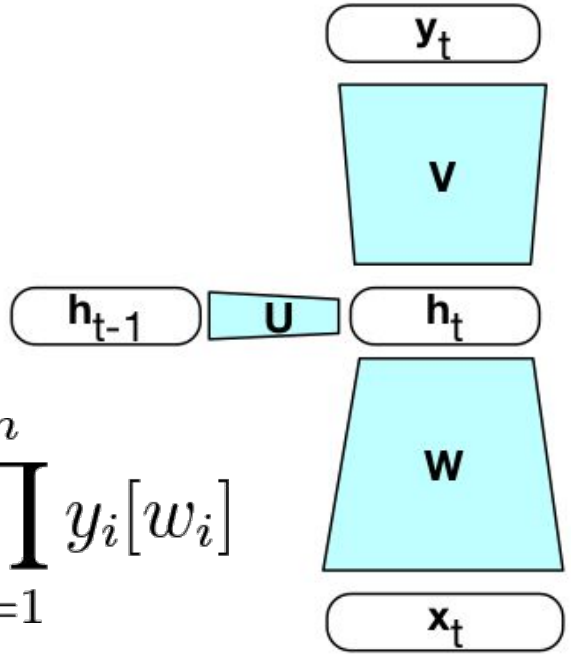
La salida en el tiempo *t* es la distribución de probabilidad de la posible siguiente palabra *t+1*



# RNN como Modelo de Lenguaje

La salida en el tiempo *t* es la distribución de probabilidad de la posible siguiente palabra *t+1*

$$P(w_{t+1} = i | w_1, ..., w_n) = y_t[i]$$




*La probabilidad de toda una secuencia:*

$$P(w_{1:n}) = \prod_{i=1}^{n} P(w_i | w_{1:i-1}) = \prod_{i=1}^{n} y_i[w_i]$$

# Entrenamiento

- Autosupervisado: usamos oraciones de un corpus conocido y no necesitamos etiquetas
- Se busca minimizar el error de predecir la siguiente palabra dadas todas las anteriores

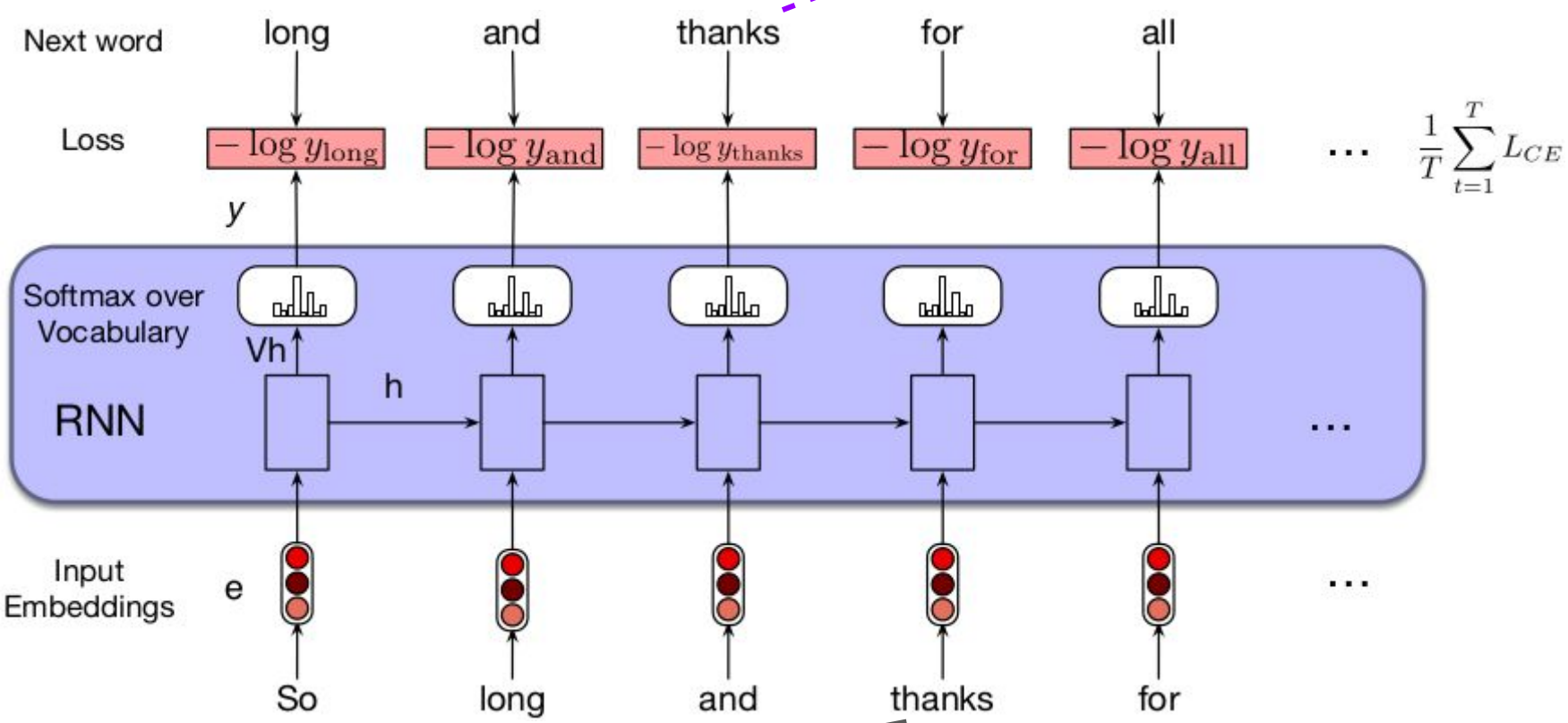
$$L_{CE} = -\sum_{w \in V} y_t[w] \log \hat{y}_t[w]$$

- $y_t$ (la distribución esperada) es un one-hot donde solo la siguiente palabra correcta vale 1

$$L_{CE} = -\log \hat{y}_t[w_{t+1}]$$

# Entrenamiento

La palabra predicha podría ser la correcta, o no




Pero durante el entrenamiento siempre se presenta la correcta a continuación *(teacher forcing)*