Introducción a las redes neuronales: Redes Recurrentes

M. Sc. Saúl Calderón Ramírez
Instituto Tecnológico de Costa Rica,
Escuela de Computación, bachillerato en Ingeniería en Computación,
PAttern Recongition and MAchine Learning Group (PARMA-Group)

13 de mayo de 2019

Resumen

Este material está basado en el capítulo de redes recurrentes, del libro *Deep Learning* de Bengio et. al. [1].

1 Redes recurrentes

Las redes recurrentes (*Rumelhart et al., 1986a*) son una familia de redes neuronales diseñadas para procesar datos secuenciales, a diferencia de las redes convolucionales, las cuales tienen por objetivo procesar matrices de valores X. Las secuencias se pueden modelar de manera generalizada como una serie de arreglos $\overrightarrow{x}^{(1)}, \overrightarrow{x}^{(2)}, \dots \overrightarrow{x}^{(\tau)}$, donde a diferencia de las matrices estudiadas anteriormente con una correlación espacial de sus entradas, las secuencias presentan una correlación entre los arreglos previos y posteriores. Las redes recurrentes proponen el procesamiento de secuencias con un largo τ arbitrario, implementando el concepto de **parámetros compartidos**, tal como se implementa en las redes convolucionales. Imagine el problema de reconocer la vocalización de la letra "a" en una grabación. Si en una grabación Y_1, la pronunciación fue capturada en 50 muestras al final de la grabación, y en otra grabación Y, la pronunciación de la letra "a" tarda 25 muestras al inicio de la grabación, el modelo debe de igual forma reconocer tal grabación como la pronuncaición de la letra "a", sin importar cuanto tardó y la posición de la misma. **Un perceptrón multicapa hubiese necesitado una longitud fija a la entrada**, y además tendería a "aprender" la posición en la secuencia.

Una secuencia puede utilizarse para representar el **estado de un sistema en distintos momentos**, mediante los arreglos $\overrightarrow{h}^{(1)}, \overrightarrow{h}^{(2)}, \dots \overrightarrow{h}^{(\tau)}$, para modelar y estimar el comportamiento de un sistema dinámico:

$$\overrightarrow{h}^{(t)} = f\left(\overrightarrow{h}^{(t-1)}, \overrightarrow{\theta}\right)$$

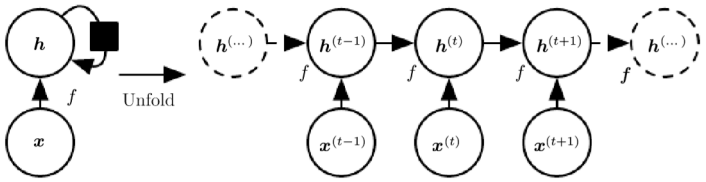


Figura 1: Red recurrente sencilla, tomado de [1].

donde $\overrightarrow{\theta}$ corresponde a los **parámetros que definen el comportamiento del sistema** en función del estado anterior.

La **relación de recurrencia** en este modelo basado en estados, se observa al reescribir la ecuación anterior como:

$$\overrightarrow{h}^{(t)} = f\left(f\left(\overrightarrow{h}^{(t-2)}, \overrightarrow{\theta}\right), \overrightarrow{\theta}\right)$$

por lo que por ejemplo, si **desplegamos** todos los **llamados recursivos** o las relaciones de recurrencia para una secuencia de largo $\tau = 3$, el estado $\overrightarrow{s}^{(3)}$ viene dado como:

$$\overrightarrow{h}^{(3)} = f\left(\overrightarrow{h}^{(2)}, \overrightarrow{\theta}\right) = f\left(f\left(\overrightarrow{h}^{(1)}, \overrightarrow{\theta}\right), \overrightarrow{\theta}\right)$$

Para generalizar aún más la notación de sistemas basados en estados, se modela también la interacción del sistema con una **señal de control** $\overrightarrow{x}^{(t)}$, la cual influye en el estado nuevo $\overrightarrow{h}^{(t)}$, expresándose como sigue:

$$\overrightarrow{h}^{(t)} = f\left(\overrightarrow{h}^{(t-1)}, \overrightarrow{x}^{(t)}, \overrightarrow{\theta}\right)$$

En términos de la nomenclatura usada para redes recurrentes, los **estados del sistema se representan en las unidades ocultas** $\overrightarrow{h}^{(t)}$, por lo que además es usual en las arquitecturas de las redes recurrentes **agregar unidades de salida** las cuales pasan por una **función de activación específica** la salida de las unidades ocultas $\overrightarrow{h}^{(t)}$. La Figura 1 muestra el ***despliegue*** de una red recurrente simple, sin unidades de salida. Observe además que **el grafo desplegado** presenta un **tamaño dependiente a la longitud específica de la secuencia**. El recuadro negro en tal figura indica una interacción con un retraso de una unidad en la secuencia (puede corresponder a una unidad de tiempo).

Un ejemplo de aplicación de una red recurrente es la predicción de nuevas palabras dada una secuencia de las mismas $\overrightarrow{x}^{(t)}$, donde $\overrightarrow{h}^{(t)}$ representa el estado *oculto* del sistema, y los parámetros a aprender $\overrightarrow{\theta}$ en la función f definen el peso o importancia de palabras anteriores para estimar el nuevo estado.

Parámetros compartidos: La Figura 1 muestra como los parámetros en el arreglo $\overrightarrow{\theta}$ definen el valor del estado actual basado únicamente en el estado anterior y la entrada actual al sistema. Esto hace posible que el modelo pueda

computar la salida para secuencias de largo τ arbitrario, con menos parámetros, necesitando entonces menos muestras para entrenar el modelo, lo cual queda claro en la siguiente equivalencia:

$$\overrightarrow{h}^{(t)} = f\left(\overrightarrow{h}^{(t-1)}, \overrightarrow{x}^{(t)}, \overrightarrow{\theta}\right) = g^{(t)}\left(\overrightarrow{x}^{(t)}, \overrightarrow{x}^{(t-1)}, \ldots, \overrightarrow{x}^{(2)}, \overrightarrow{x}^{(1)}\right)$$

Redes recurrentes como mezcla de una red neuronal y un modelo oculto de Markov: Las redes recurrentes típicas agregan los siguientes componentes, usuales en las arquitecturas completamente conectadas y convolucionales, basada en el concepto de estado oculto, utilizado en las redes ocultas de Markov:

1. **Unidades de entrada** $\overrightarrow{x}^{(t)} \in \mathbb{R}^D$: La unidad de entrada corresponde a la secuencia de valores usada para realizar la estimación de la salida, y su conexión con las unidades ocultas $\overrightarrow{h}^{(t)}$se representa en la matriz $U \in \mathbb{R}^{D \times M}$.

2. **Unidades ocultas** $\overrightarrow{h}^{(t)} \in \mathbb{R}^M$: La unidad oculta en una casilla anterior de la secuencia $t-1$ modela una relación con la unidad oculta en la casilla actual t a través de la matriz de parámetros $W \in \mathbb{R}^{M \times M}$. La salida de las unidades ocultas se computa en el modelo presentado en la Figura 2, de la siguiente forma, basado en la combinación lineal de la secuencia de entrada actual y el estado anterior:

   $$\overrightarrow{p}^{(t)} = \overrightarrow{b} + W'\overrightarrow{h'}^{(t-1)} + U'\overrightarrow{x'}^{(t)} = W\overrightarrow{h}^{(t-1)} + U\overrightarrow{x}^{(t)}$$

   para luego pasar tal **peso neto** $\overrightarrow{p}^{(t)}$ por la **función de activación**, como por ejemplo, el tangente hiperbólico, en este caso **con salida vectorial para simplificar la notación, lo que denominaremos como notación vectorial** $g^o\left(\overrightarrow{p}^{(t)}\right) = \tanh\left(\overrightarrow{p}^{(t)}\right)$:

   $$\overrightarrow{h}^{(t)} = g^h\left(\overrightarrow{p}^{(t)}\right) = g^h\left(W\overrightarrow{h}^{(t-1)} + U\overrightarrow{x}^{(t)}\right)$$

3. **Unidades de salida** $\overrightarrow{o}^{(t)} \in \mathbb{R}^K$: La asociación entra las unidades de salida $\overrightarrow{o}^{(t)}$ y las unidades ocultas $\overrightarrow{h}^{(t)}$ se representa en la matriz de parámetros $V \in \mathbb{R}^{M \times K}$, computado de la siguiente forma:

   $$\overrightarrow{o}^{(t)} = \overrightarrow{c} + V'\overrightarrow{h'}^{(t)} = V\overrightarrow{h}^{(t)}.$$

   La salida estimada $\overrightarrow{\hat{y}}^{(t)}$ usualmente se computa al pasar el valor resultante de la unidad de salida $\overrightarrow{o}^{(t)}$ por una función de activación como la función *softmax* $g^s\left(\overrightarrow{o}^{(t)}\right) = \textit{softmax}\left(\overrightarrow{o}^{(t)}\right)$, de forma que:

   $$\overrightarrow{\hat{y}}^{(t)} = g^o\left(\overrightarrow{o}^{(t)}\right) = g^o\left(V\overrightarrow{h}^{(t)}\right)$$

4. **Función de pérdida** $L^{(t)} \in \mathbb{R}$ **y** ***groundtruth*** $\overrightarrow{y}^{(t)} \in \mathbb{R}^K$**:** La función de pérdida mide *que tan lejos* está la predicción $\overrightarrow{\hat{y}}^{(t)}$ del *ground truth o target*

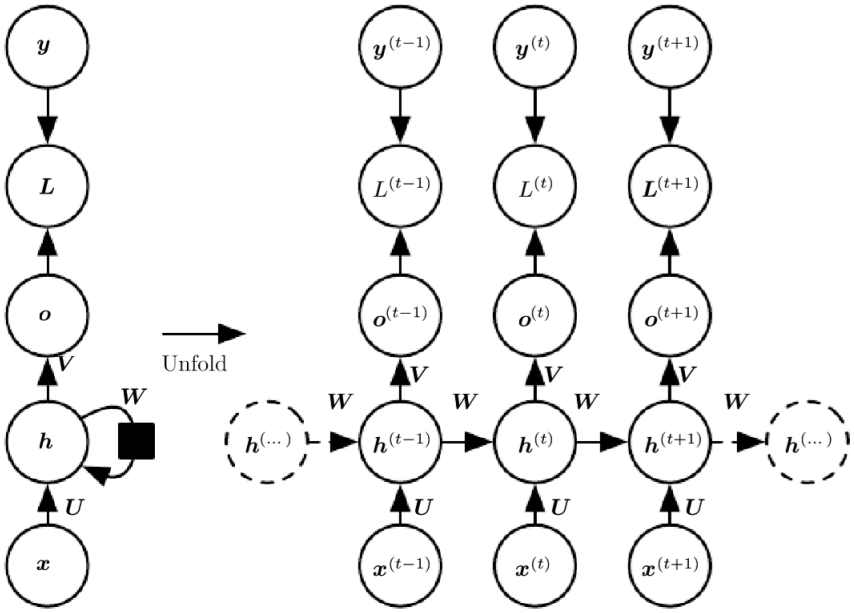


Figura 2: Red recurrente para estimar una salida por casilla (izquierda sin desenvolver, derecha desenvuelta), tomado de [1].

$\overrightarrow{y}^{(t)}$. La función de pérdida en general viene dada por:

$$L\left(\overrightarrow{\widehat{y}}, \overrightarrow{y}\right) = L\left(\left\{\overrightarrow{\widehat{y}}^{(1)}, \overrightarrow{\widehat{y}}^{(2)}, \ldots, \overrightarrow{\widehat{y}}^{(\tau)}\right\}, \left\{\overrightarrow{y}^{(1)}, \overrightarrow{y}^{(2)}, \ldots, \overrightarrow{y}^{(\tau)}\right\}\right) = \sum_t L^{(t)}$$

y en el caso específico de por ejemplo usar la **entropía cruzada** como función de pérdidad, se tiene que:

$$L\left(\overrightarrow{\widehat{y}}, \overrightarrow{y}\right) = \overrightarrow{y} \log\left(\overrightarrow{\widehat{y}}\right) = -\sum \overrightarrow{y} \log\left(\overrightarrow{\widehat{y}}\right)_t$$

Los **parámetros a estimar** corresponden a las matrices W', U', V' y los sesgos $\overrightarrow{b}$ y $\overrightarrow{c}$, o en su defecto con las matrices extendidas con el sesgo W, U y V. Recuerde además que si por ejemplo $\overrightarrow{x}^{(t)} \in \mathbb{R}^{D\times 1}$,se debe cumplir que $U \in \mathbb{R}^{M\times D}$, de forma similar para las matrices $W \in \mathbb{R}^{M\times M}$ y $V \in \mathbb{R}^{K\times M}$.

La Figura 2 muestra la arquitectura básica de una red recurrente la cual realiza una estimación por cada casilla.

La Figura 3 muestra la comparativa de una red recurrente, con un perceptrón multicapa común, usando **notación envuelta y vectorial**, donde se nota que para el caso del perceptrón multicapa $U = W^o$ y $V = W^s$, según la notación usada anteriormente para el perceptrón multicapa.

Otras variantes de la red anterior, ilustradas en la Figura 4, consisten en conectar las unidades de salida $\overrightarrow{o}^{(t)}$ con las unidades ocultas $\overrightarrow{h}^{(t)}$, o en ca-

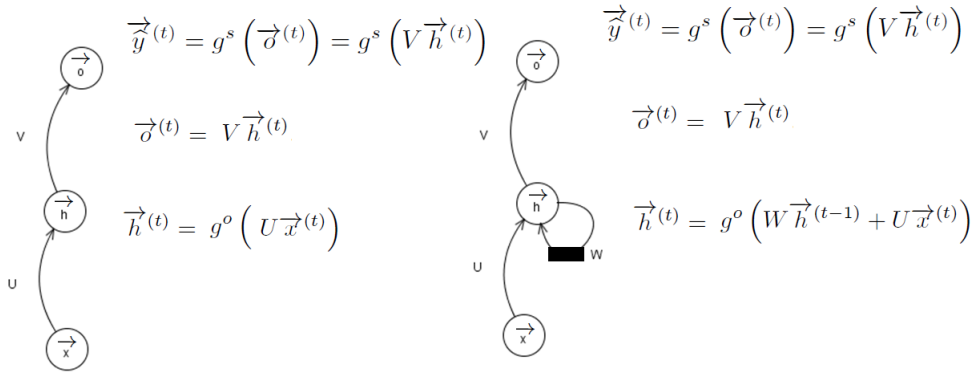


Figura 3: Perceptrón multicapa vs. red recurrente en representación *envuelta*.

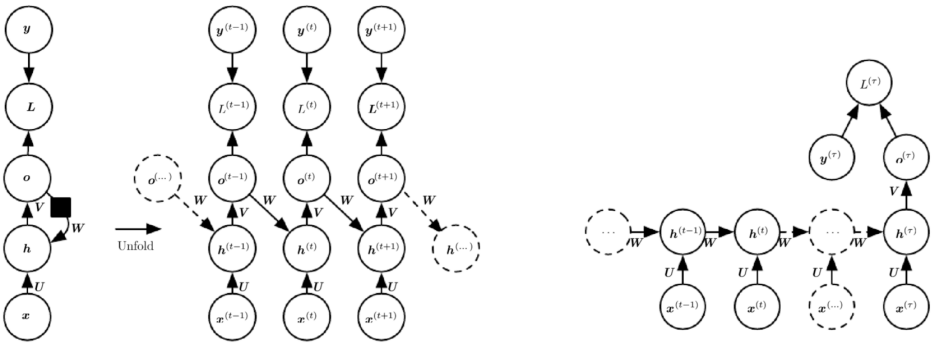


Figura 4: Variantes de arquitecturas de redes recurrentes, izquierda: alimentación desde la unidad de salida a la oculta, derecha: una sola estimación por toda la secuencia, tomado de [1].

so de ser necesario estimar un arreglo $\overrightarrow{\hat{y}}^{(\tau)}$ por toda una secuencia de valores anteriores $\left\{\overrightarrow{x}^{(t)}, \overrightarrow{x}^{(t-1)}, \ldots, \overrightarrow{x}^{(2)}, \overrightarrow{x}^{(1)}\right\}$, como en el ejemplo de estimar el lenguaje al que pertenece una palabra, donde cada caracter o letra corresponde a una casilla en la secuencia.

La variante de la izquierda en la Figura 4 es menos poderosa, pues las unidades de salida deben además de acercarse al valor deseado, capturar la información de casillas anteriores. Sin embargo, la **ventaja** de conectar las unidades de salida con las ocultas mediante la matriz W, es la posibilidad de **usar el valor conocido del *groundtruth*** $\overrightarrow{y}^{(\tau)}$ en vez de su aproximación, lo que permite calcular el gradiente por **cada casilla o unidad de tiempo de forma separada, facilitando su paralelización y computo más rápido**. Lo anterior se conoce como el **forzamiento de la enseñanza** o *teacher forcing*, y se ilustra en la Figura 5.

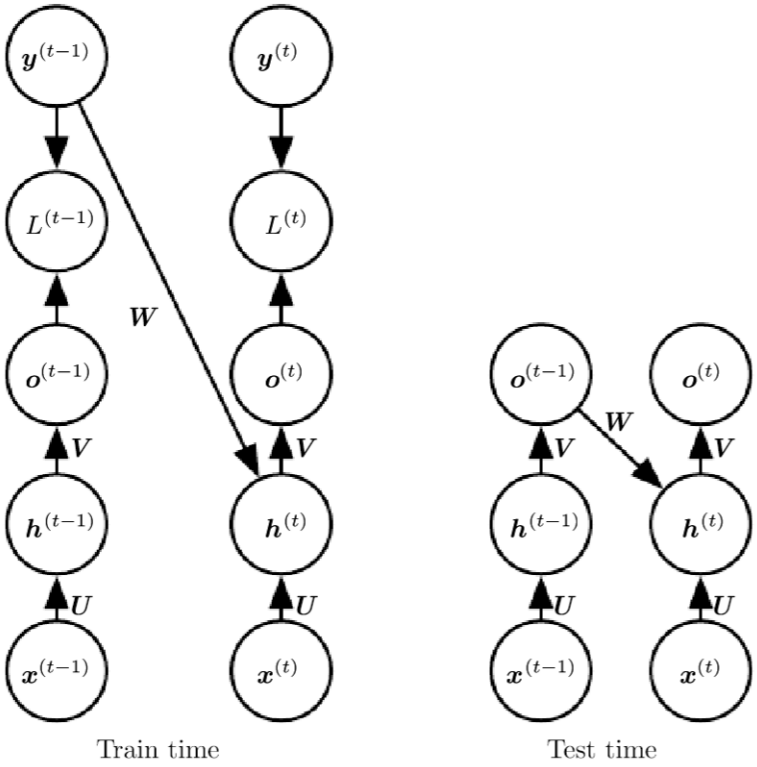


Figura 5: Forzamiento de la enseñanza: Utilizar la salida real en vez de la salida estimada para acelerar el entrenamiento. Para la estimación de la salida en tiempo de evaluación, se utiliza la salida estimada en la casilla anterior, tomado de [1].

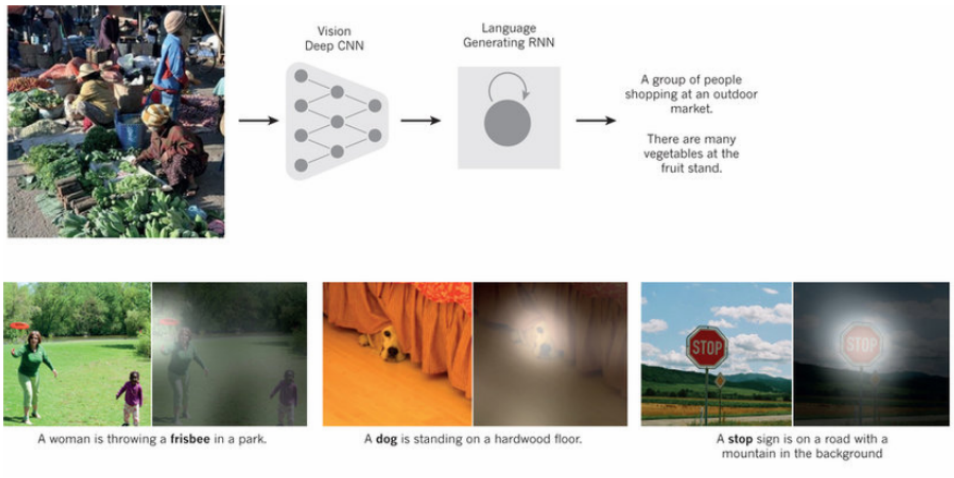


Figura 6: Subtitulos generados a partir de la imagen, junto con enfoque de atención, tomado de *Show and Tell: A Neural Image Caption Generator, 2015*.

1.1 Variantes de redes recurrente según secuencias a la entrada y/o salida

La arquitectura *vainilla* de un perceptrón multi-capa o una red completamente conectada en su notación *compacta* se muestra en la Figura 7, y recibe una secuencia de un elemento $\overrightarrow{x} \in \mathbb{R}^D$, y tiene como salida también una secuencia de un elemento $\overrightarrow{o} \in \mathbb{R}^K$, con la matriz U conectando de forma completa las unidades en la capa oculta $\overrightarrow{h}$, y la matriz V conectando tales unidades ocultas a las unidades de salida. Esta **arquitectura de uno a uno** completamente conectada es común utilizarla para por ejemplo **clasificar una imagen de un digito** $\overrightarrow{x} \in \mathbb{R}^D$ en 10 clases.

Las arquitecturas de **uno a muchos** son apropiadas para estimar una secuencia de valores $\overrightarrow{o}^{(1)}, \overrightarrow{o}^{(2)}, \dots \overrightarrow{o}^{(\tau)}$ a partir de una entrada $\overrightarrow{x} \in \mathbb{R}^D$, como por ejemplo en el problema de **estimar un subtítulo (una serie de palabras) a una imagen**, como se puede observar en la Figura 6.

Para problemas donde es necesario clasificar una secuencia de arreglos $\overrightarrow{x}^{(1)}, \overrightarrow{x}^{(2)}, \dots \overrightarrow{x}^{(\tau)}$ en una clase específica, con una sola salida $\overrightarrow{o} \in \mathbb{R}^K$, las arquitecturas **muchas a una** son usuales. Por ejemplo, la **clasificación de sentencias o secuencias de palabras en su sentimiento (positivo, neutro o negativo, a nivel de palabras)**, o por ejemplo el reconocimiento de acciones en vídeos o la predicción del siguiente caracter cuando se escribe una palabra, son problemas apropiados para tales arquitecturas.

Las arquitecturas de **muchas a muchas** reciben de entrada una secuencia de vectores $\overrightarrow{x}^{(1)}, \overrightarrow{x}^{(2)}, \dots \overrightarrow{x}^{(\tau)}$ y estiman una serie de valores $\overrightarrow{o}^{(1)}, \overrightarrow{o}^{(2)}, \dots \overrightarrow{o}^{(\tau)}$, típicas para resolver problemas de por ejemplo **traducción de textos**, dónde las secuencias pueden tener largos distintos, o de **rastreo de objetos o estima-**

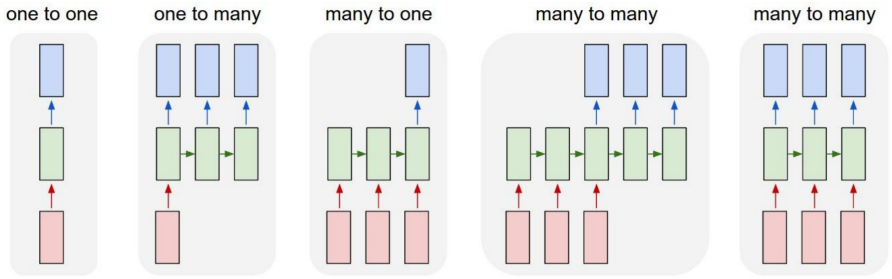


Figura 7: Variantes de una red recurrente, con notación compacta pero desenvuelta, tomado de *Stanford University School of Engineering* `http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf`.

ción de emociones de cuadro a cuadro en un vídeo, dónde los largos de las secuencias son iguales. La predicción del siguiente caracter cuando se escribe una palabra también se puede implementar con esta arquitectura, como se ilustra en la Figura 8. En **tiempo de entrenamiento**, la pérdida se compara la salida del modelo con el valor deseado en la entrada de esa secuencia.

Existen problemas que se pueden abordar con distintas arquitecturas, por ejemplo la clasificación de imágenes de dígitos se puede enfocar como la clasificación de una serie de pixeles de la imagen esqueletonizada, usando una red recurrente de muchos a uno.

Una variante típica en entornos de traducción de textos son las **redes recurrentes codificadoras-decodificadoras**, las cuales implementan primero una etapa de muchos a uno, para codificar la secuencia $\overrightarrow{x}^{(1)}, \overrightarrow{x}^{(2)}, \ldots \overrightarrow{x}^{(\tau)}$ en un sólo vector $\overrightarrow{o} \in \mathbb{R}^K$, lo cual **permite comprimir o descubrir información latente en la secuencia** y luego implementar una etapa de decodificación con una a arquitectura uno a muchos, como lo ilustra la Figura 9.

1.2 Computo del gradiente del error para redes recurrentes

Para computar el gradiente en una red recurrente se debe **desenvolver el grafo** con todas las casillas de la secuencia, o analogamente en todas las unidades de tiempo de la secuencia, por lo que a este proceso se llama **propagación hacia atrás a través del tiempo** o BPTT por sus siglas en inglés. La Figura 3 tomada de

Para realizar la propagación hacia atrás, por cada casilla t en la secuencia (iniciando de $t = 0$ hasta el largo), se debe primero realizar la **pasada hacia adelante.**

Es necesario entonces computar el gradiente respecto a las matrices $V \in \mathbb{R}^{K\times M}$,$W \in \mathbb{R}^{M\times M}$ y $U \in \mathbb{R}^{M\times D}$, para actualizar sus pesos siguiendo la ecuación

$$w(\eta + 1) = w(\eta) + \alpha \Delta w(\eta),$$

para una iteración η.

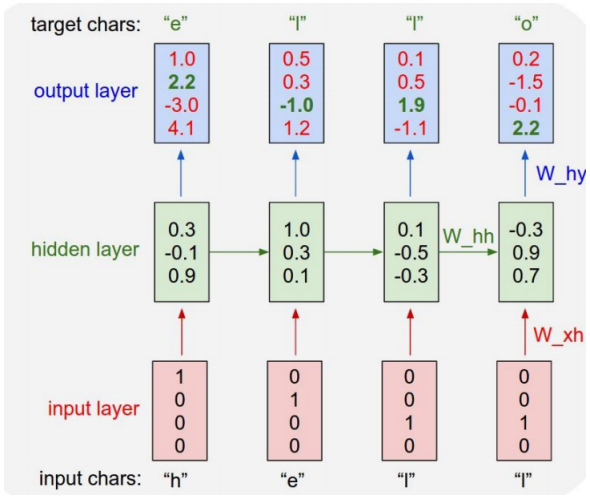


Figura 8: Estimación del caracter siguiente, a partir de una secuencia de caracteres, con una arquitectura de muchas a muchas, con igual largo de secuencia tomado de *Stanford University School of Engineering* `http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf`.

Siguiendo el ejemplo de la red propuesta en la sección anterior, ilustrada en la Figura 2, donde para la unidad oculta se define:

$$\overrightarrow{h}^{(t)} = f^h\left(\overrightarrow{p}^{(t)}\right) = f^h\left(W\overrightarrow{h}^{(t-1)} + U\overrightarrow{x}^{(t)}\right)$$

y para una unidad oculta en particular m:

$$h_m^{(t)} = f^h\left(p_m^{(t)}\right) = f^h\left(W_{m,:}\overrightarrow{h}^{(t-1)} + U_{m,:}\overrightarrow{x}^{(t)}\right)$$

de formacon $\overrightarrow{p}^{(t)} = W\overrightarrow{h}^{(t-1)} + U\overrightarrow{x}^{(t)}$, con lo que según lo comentado anteriormente (abusando de la notación vectorial):

$$\nabla_{\overrightarrow{p}^{(t)}} f^h\left(\overrightarrow{p}^{(t)}\right)$$

y para la unidad de salida:

$$\overrightarrow{o}^{(t)} = V\overrightarrow{h}^{(t)}$$

para una unidad de salida k:

$$o_k^{(t)} = V_{k,:}\overrightarrow{h}^{(t)}$$

finalmente la salida del modelo está entonces dada por:

$$\overrightarrow{\hat{y}}^{(t)} = f^o\left(\overrightarrow{o}^{(t)}\right)$$

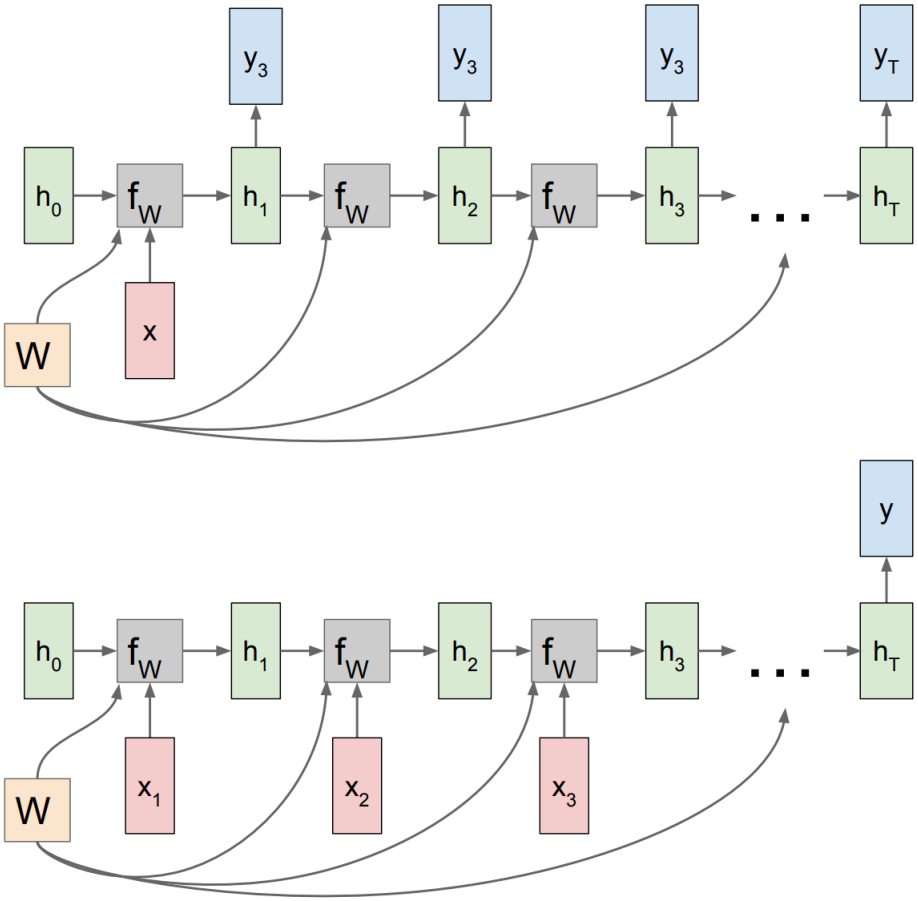


Figura 9: Arquitectura codificadora-decodificadora, tomado de *Stanford University School of Engineering* `http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf`.

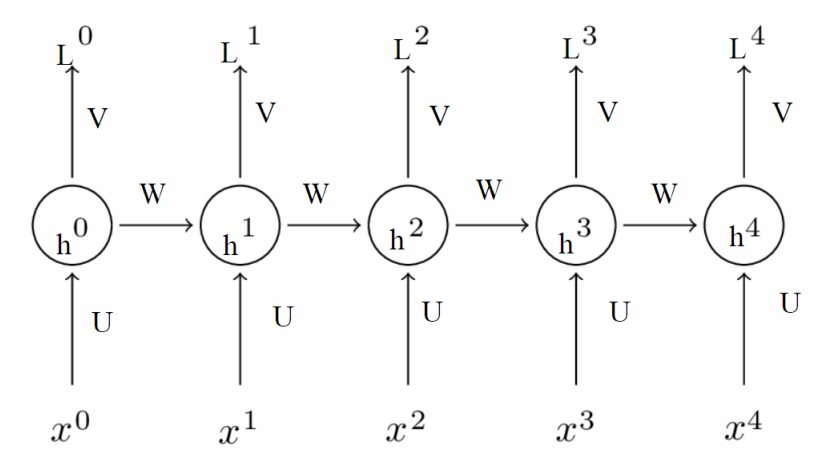


Figura 10: Grafo desenvuelto para muestra $n = 1$, con largo $\tau = 4$.

La **función de pérdida para una casilla** t de la secuencia se define entonces como sigue (ejemplificandose por ejemplo con una función de pérdida euclidiana), de forma que para una casilla t:

$$L^{(t)} = \left\| \overrightarrow{\widehat{y}}^{(t)} - \overrightarrow{y}^{(t)} \right\|^2 = \frac{1}{2} \sum_{k}^{K} \left(\widehat{y}_k^{(t)} - y_k^{(t)} \right)^2$$

Y la **función de pérdida total** vendría a definirse como la acumulación de la pérdida para todas las casilla:

$$L = \sum_{t}^{\tau} L^{(t)}$$

Asuma ahora que el gradiente se va calcular para una muestra $n = 1$, de largo $\tau = 4$, como la ilustrada en la Figura 10.

La Figura 11 ilustra un caso donde se tiene una entrada de dimensión $D = 1$, con $M = 3$ unidades ocultas y $K = 1$ unidad de salida.

1. **Gradiente para la capa de salida**. Posterior al cómputo de la pasada hacia adelante, se computa el gradiente desde la función de pérdida hasta la capa de entrada, por cada unidad t. Observe según el grafo 10, que la matriz $V \in \mathbb{R}^{K \times M}$ representa el arista entre la pérdida y la unidad oculta, por lo que entonces se calcula:

$$\frac{\partial L^{(3)}}{\partial V_{k,m}} = \frac{\partial L^{(3)}}{\partial \widehat{y}_k^{(3)}} \frac{\partial \widehat{y}_k^{(3)}}{\partial o_k^{(3)}} \frac{\partial o_k^{(3)}}{\partial V_{k,m}}$$

   lo cual indica que **sólo es necesario el valor actual en la secuencia, y no los anteriores**. Lo que es exactamente idéntico al cálculo de gradiente

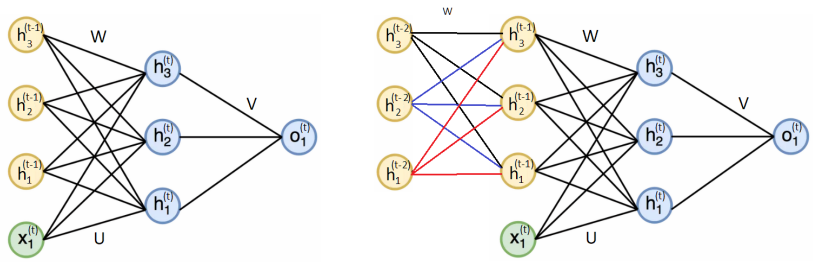


Figura 11: Red recurrente desenvuelta para 3 instantes de tiempo.

para el perceptrón multicapa, en la capa de salida. De forma similar, para reutilizar los primeros dos términos en cálculos sucesivos donde varía la unidad oculta m, se define el δ_k^s como sigue:

$$\delta_k^{v^{(3)}} = \frac{\partial L^{(3)}}{\partial \widehat{y}_k^{(3)}} \frac{\partial \widehat{y}_k^{(3)}}{\partial o_k^{(3)}}$$

por lo que:

$$\frac{\partial L^{(3)}}{\partial V_{k,m}} = \delta_k^{v^{(3)}} \frac{\partial o_k^{(3)}}{\partial V_{k,m}}.$$

Los pesos entonces de la matriz V se actualizan de la siguiente forma:

$$V_{k,m}\left(\eta+1\right) = V_{k,m}\left(\eta\right) - \alpha \triangle V_{k,m}\left(\eta\right), \tag{1}$$

$$\triangle V_{k,m}\left(\eta\right) = \delta_k^{v^{(3)}} h_m^{(3)}.$$

1. **Gradiente en la capa oculta**: Para calcular el gradiente en la capa oculta, se estima el cambio para las matrices de parámetros $W \in \mathbb{R}^{M\times M}$ y $U \in \mathbb{R}^{M\times D}$. Veamos primero el desarrollo de la relación de recurrencia para los otros términos anteriores, en las neuronas ocultas:

$$\overrightarrow{\widehat{y}}^{(t)} = f^o\left(\overrightarrow{o}^{(t)}\right)$$

$$\overrightarrow{o}^{(t)} = V\overrightarrow{h}^{(t)}$$

$$\overrightarrow{h}^{(t)} = f^h\left(\overrightarrow{p}^{(t)}\right) = f^h\left(W\overrightarrow{h}^{(t-1)} + U\overrightarrow{x}^{(t)}\right)$$

$$\overrightarrow{h}^{(t)} = f^h\left(Wf^h\left(W\overrightarrow{h}^{(t-2)} + U\overrightarrow{x}^{(t-1)}\right) + U\overrightarrow{x}^{(t)}\right)$$

$$\overrightarrow{h}^{(t)} = f^h\left(Wf^h\left(Wf^h\left(W\overrightarrow{h}^{(t-3)} + U\overrightarrow{x}^{(t-2)}\right) + U\overrightarrow{x}^{(t-1)}\right) + U\overrightarrow{x}^{(t)}\right)$$

$$\overrightarrow{h}^{(t)} = f^h\left(Wf^h\left(Wf^h\left(W\left(U\overrightarrow{x}^{(t-3)}\right) + U\overrightarrow{x}^{(t-2)}\right) + U\overrightarrow{x}^{(t-1)}\right) + U\overrightarrow{x}^{(t)}\right)$$

Con lo cual, para $t = 3$

$$\overrightarrow{h}^{(3)} = f^h\left(Wf^h\left(Wf^h\left(W\left(U\overrightarrow{x}^{(0)}\right) + U\overrightarrow{x}^{(1)}\right) + U\overrightarrow{x}^{(2)}\right) + U\overrightarrow{x}^{(3)}\right) \quad (2)$$

Se puede observar facilmente que el gradiente tanto para U como para W depende de la relación recurrente. Examinamos primero la derivada parcial respecto a los pesos que conectan la capa de entrada con las unidades ocultas $U \in \mathbb{R}^{M\times D}$:

$$\frac{\partial L^{(3)}}{\partial U_{m,d}} = \sum_{k=0}^{K}\left(\frac{\partial L^{(3)}}{\partial \widehat{y}_k^{(3)}}\frac{\partial \widehat{y}_k^{(3)}}{\partial o_k^{(3)}}\frac{\partial o_k^{(3)}}{\partial h_m^{(3)}}\right)\frac{\partial h_m^{(3)}}{\partial p_m^{(3)}}\frac{\partial p_m^{(3)}}{\partial h_m^{(2)}}\frac{\partial h_m^{(2)}}{\partial p_m^{(2)}}\frac{\partial p_m^{(2)}}{\partial h_m^{(1)}}\frac{\partial h_m^{(1)}}{\partial p_m^{(1)}}\frac{\partial p_m^{(1)}}{\partial h_m^{(0)}}\frac{\partial h_m^{(0)}}{\partial p_m^{(0)}}\frac{\partial p_m^{(0)}}{\partial U_{m,d}} \quad (3)$$

Ahora analicemos los términos independientes de las funciones de activación, para un tiempo cualquiera t:

$$\frac{\partial h_m^{(t)}}{\partial p_m^{(t)}} = \frac{\partial}{\partial p_m^{(t)}} f^h\left(p_m^{(t)}\right) \quad (4)$$

$$\frac{\partial p_m^{(t)}}{\partial h_m^{(t-1)}} = \frac{\partial}{\partial h_m^{(t-1)}}\left(W_{m,:}\overrightarrow{h}^{(t-1)} + U_{m,:}\overrightarrow{x}^{(t)}\right) = W_{m,m} \quad (5)$$

$$\frac{\partial p_m^{(0)}}{\partial U_{m,d}} = x_d^{(0)}$$

y en general, para cualquier t:

$$\frac{\partial p_m^{(t)}}{\partial U_{m,d}} = x_d^{(t)}$$

con lo que entonces podemos escribir el término más compacto, juntando las ecuaciones 4 y 5:

$$\delta_m^{u^{(t)}} = \frac{\partial h_m^{(t)}}{\partial h_m^{(t-1)}} = \frac{\partial}{\partial p_m^{(t)}} f^h\left(p_m^{(t)}\right) W_{m,m}$$

y en general, para diferencias de tiempo de largo $k > 0$ arbitrarias:

$$\frac{\partial h_m^{(t)}}{\partial h_m^{(t-k)}} = \prod_{j=0}^{k-1}\frac{\partial}{\partial p_m^{(t-j)}} f^h\left(p_m^{(t-j)}\right)\left(W_{m,m}\right)^k$$

lo cual en términos del delta $\delta_m^{u^{(t)}}$ se puede expresar de forma más simple como:

$$\frac{\partial h_m^{(t)}}{\partial h_m^{(t-k)}} = \delta_m^{u^{(t)}}\delta_m^{u^{(t-1)}}\cdots\delta_m^{u^{(t-k)}}$$

Por lo que la ecuación 3 se puede reescribir como:

$$\frac{\partial L^{(3)}}{\partial U_{m,d}} = \sum_{k=0}^{K} \left(\frac{\partial L^{(3)}}{\partial \widehat{y}_k^{(3)}} \frac{\partial \widehat{y}_k^{(3)}}{\partial o_k^{(3)}} \frac{\partial o_k^{(3)}}{\partial h_m^{(3)}} \right) \delta_m^{u^{(3)}} \delta_m^{u^{(2)}} \delta_m^{u^{(1)}} \frac{\partial h_m^{(0)}}{\partial p_m^{(0)}} \frac{\partial p_m^{(0)}}{\partial U_{m,d}} \tag{6}$$

$$\frac{\partial L^{(3)}}{\partial U_{m,d}} = \sum_{k=0}^{K} \left(\frac{\partial L^{(3)}}{\partial \widehat{y}_k^{(3)}} \frac{\partial \widehat{y}_k^{(3)}}{\partial o_k^{(3)}} \frac{\partial o_k^{(3)}}{\partial h_m^{(3)}} \right) \frac{\partial h_m^{(3)}}{\partial h_m^{(3-3)}} \frac{\partial h_m^{(0)}}{\partial p_m^{(0)}} \frac{\partial p_m^{(0)}}{\partial U_{m,d}} \tag{7}$$

Sin embargo observe que en esta derivación parcial se tomó en cuenta únicamente la contribución del gradiente para la última aparición de la matriz $U_{m,d}$ en la ecuación 2, por lo que dado que los pesos son compartidos, se suma la contribución de los gradientes para los otros términos de la siguiente forma:

$$\frac{\partial L^{(3)}}{\partial U_{m,d}} = \sum_{k=0}^{K} \left(\frac{\partial L^{(3)}}{\partial \widehat{y}_k^{(3)}} \frac{\partial \widehat{y}_k^{(3)}}{\partial o_k^{(3)}} \frac{\partial o_k^{(3)}}{\partial h_m^{(3)}} \right) \sum_{\tau=0}^{3-1} \frac{\partial h_m^{(3)}}{\partial h_m^{(\tau)}} \frac{\partial h_m^{(\tau)}}{\partial p_m^{(\tau)}} \frac{\partial p_m^{(\tau)}}{\partial U_{m,d}}$$

y en general

$$\frac{\partial L^{(t)}}{\partial U_{m,d}} = \sum_{k=0}^{K} \left(\frac{\partial L^{(t)}}{\partial \widehat{y}_k^{(t)}} \frac{\partial \widehat{y}_k^{(t)}}{\partial o_k^{(t)}} \frac{\partial o_k^{(t)}}{\partial h_m^{(t)}} \right) \sum_{\tau=0}^{t-1} \frac{\partial h_m^{(t)}}{\partial h_m^{(\tau)}} \frac{\partial h_m^{(\tau)}}{\partial p_m^{(\tau)}} \frac{\partial p_m^{(\tau)}}{\partial U_{m,d}}$$

$$\frac{\partial L^{(t)}}{\partial U_{m,d}} = \sum_{k=0}^{K} \left(\frac{\partial L^{(t)}}{\partial \widehat{y}_k^{(t)}} \frac{\partial \widehat{y}_k^{(t)}}{\partial o_k^{(t)}} \frac{\partial o_k^{(t)}}{\partial h_m^{(t)}} \right) \sum_{\tau=0}^{t-1} \delta_m^{u^{(t)}} \delta_m^{u^{(t-1)}} \cdots \delta_m^{u^{(\tau)}} \frac{\partial h_m^{(\tau)}}{\partial p_m^{(\tau)}} \frac{\partial p_m^{(\tau)}}{\partial U_{m,d}}$$

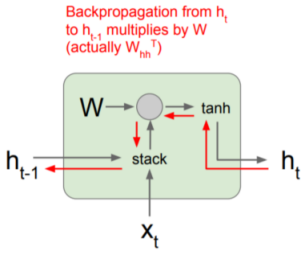
definiendo entonces la ecuación de actualización del gradiente para la matriz $U \in \mathbb{R}^{M \times D}$ de la forma:

$$U_{m,d}(\eta + 1) = U_{m,d}(\eta) - \alpha \triangle U_{m,d}(\eta), \tag{8}$$

$$\triangle U_{m,d}(\eta) = \frac{\partial L^{(3)}}{\partial U_{m,d}}.$$

Para el caso del gradiente respecto a la matriz $W \in \mathbb{R}^{M \times M}$, sucede algo muy similar, la propagación se realiza a través del tiempo, y no de las capas, y se calcula el delta por cada instante de tiempo, basado en las conexiones del instante posterior, puesto que como lo ilustra la Figura 11, el cambio para una unidad oculta específica para la matriz de pesos W se influye por las conexiones de la capa de unidades en tal instante posterior. Para la primer capa (del instante $t = 3$ en este caso), se tiene que

$$\frac{\partial L^{(3)}}{\partial W_{m_1,m_2}} = \sum_{k=0}^{K} \left(\frac{\partial L^{(3)}}{\partial \widehat{y}_k^{(3)}} \frac{\partial \widehat{y}_k^{(3)}}{\partial o_k^{(3)}} \frac{\partial o_k^{(3)}}{\partial h_{m_1}^{(3)}} \right) \sum_{\tau=0}^{3-1} \frac{\partial h_{m_1}^{(3)}}{\partial h_{m_1}^{(\tau)}} \frac{\partial h_{m_1}^{(\tau)}}{\partial p_{m_1}^{(\tau)}} \frac{\partial p_{m_1}^{(\tau)}}{\partial W_{m_1,m_2}}$$



$$
\begin{aligned}
h_t &= \tanh(W_{hh}h_{t-1} + W_{xh}x_t) \\
&= \tanh\left((W_{hh} \quad W_{hx})\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right) \\
&= \tanh\left(W\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)
\end{aligned}
$$

Figura 12: Pasada hacia adelante y hacia atrás en una red recurrente, tomado de `http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf`.

donde la derivada parcial para $t > 0$ está dada por:

$$\frac{\partial p_{m_1}^{(t)}}{\partial W_{m_1,m_2}} = h_{m_1}^{(t-1)}$$

de esta forma la actualización del peso específico W_{m_1,m_2} viene dada por la **acumulación o agregación del gradiente** en todos los instantes de tiempo:

$$\triangle W_{m_1,m_2}(\eta) = \frac{\partial L^{(3)}}{\partial W_{m_1,m_2}}.$$

Donde para este caso el largo de la secuencia es $\tau = 3$.

La Figura 17 ilustra el flujo de la pasada hacia adelante (negro), y la pasada hacia atrás en una celda de una red recurrente.

Observe que entre más larga la secuencia, mayor cantidad de cálculos son necesarios, y además, los cálculos de los $\delta_m^{u^{(\tau)}}$ se pueden reutilizar, fluyendo de atra´s hacia adelante.

1.2.1 Propagación hacia atrás truncada

Muchas veces las secuencias a procesar en una red recurrente pueden ser muy largas, por lo que computar la pérdida para todas las entradas de la secuencia puede llegar a ser muy costoso. Es por ello que la **propagación hacia atrás en tales casos se implementa muchas veces en sub-secuencias**, para evitar la evaluación en toda la secuencia. Sin embargo los valores de las unidades ocultas pueden guardarse para las pasadas en otras porciones de la secuencia, como se ilustra en la Figura 13. De esta forma se crean lotes de usualmente $\tau = 100$ unidades de tiempo o componentes de la secuencia.

1.3 El desvanecimiento del gradiente

Además del problema del costo de computar el gradiente para secuencias muy largas, existe lo que se conoce como el desvanecimiento del gradiente. Observe

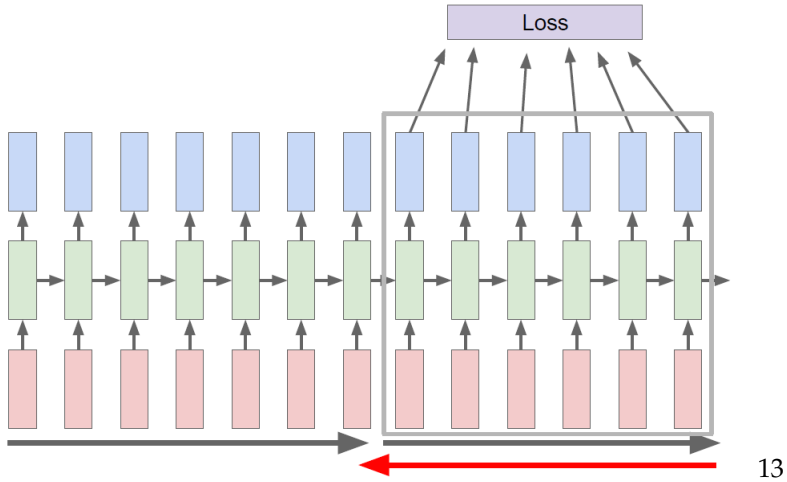


Figura 13: Propagación truncada hacia atrás, tomado de *Stanford University School of Engineering* `http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf`.

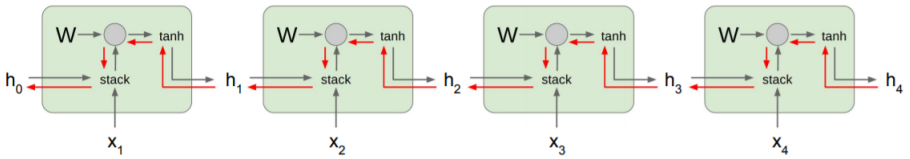


Figura 14: Flujo del gradiente desde las últimas celdas a las primeras, tomado de `http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf`.

la Figura 14. El computo del gradiente para $\tau = 4$ elementos en la secuencia involucra el flujo del gradiente del elemento 4 al elemento 1 como se puede apreciar. Esto implica cómputos en serie, que pueden realizarse arbitrariamente profundos para las unidades ocultas, las cuales al ser recurrentes, se conectan con las unidades ocultas anteriores.

Ejemplo Tomemos un ejemplo concreto, donde la red recurrente tiene por objetivo, a partir de una secuencia de entrada que representa el porcentaje de población en desempleo de Costa Rica en un mes, se desea estimar el porcentaje de confianza el siguiente mes del banco central en la economía. Examinaremos una porción de 3 valores en la secuencia:

$$\overrightarrow{x}^{(0)} = 0{,}3,\ \overrightarrow{x}^{(1)} = 0{,}35,\ \overrightarrow{x}^{(2)} = 0{,}4$$
$$\overrightarrow{y}^{(0)} = 0{,}8,\ \overrightarrow{y}^{(1)} = 0{,}81,\ \overrightarrow{y}^{(2)} = 0{,}62$$

con la siguiente arquitectura de red:

$$h_m^{(t)} = f^h\left(p_m^{(t)}\right) = f^h\left(W_{m,:}\overrightarrow{h}^{(t-1)} + U_{m,:}\overrightarrow{x}^{(t)}\right)$$

con la función de activación $f^h(x) = \mathrm{x} \Rightarrow \frac{d}{dx}f^h(x) = 1$

Y respecto a la salida neta y estimada del modelo:

$$o_k^{(t)} = V_{k,:}\overrightarrow{h}^{(t)}$$

$$\overrightarrow{\widehat{y}}^{(t)} = f^o\left(\overrightarrow{o}^{(t)}\right)$$

con la función de activación $f^o(x) = x \Rightarrow \frac{d}{dx}f^o(x) = 1$. Finalmente la función de pérdida escogida es la distancia Euclidiana.

$$L^{(t)} = \left\|\overrightarrow{\widehat{y}}^{(t)} - \overrightarrow{y}^{(t)}\right\|^2 = \frac{1}{2}\sum_k^K\left(\widehat{y}_k^{(t)} - y_k^{(t)}\right)^2$$

Se definen $M = 3$ neuronas en la capa oculta, $D = 1$ dimensiones a la entrada y $K = 1$ dimensiones a la salida. Las matrices tienen entonces las dimensiones $U \in \mathbb{R}^{3\times1}$,$W \in \mathbb{R}^{3\times3}$ y $V \in \mathbb{R}^{1\times3}$, las cuales se inicializan de la siguiente forma:

$$U(1) = \begin{bmatrix}0{,}5\\0{,}5\\0{,}5\end{bmatrix}\ W(1) = \begin{bmatrix}0{,}5 & 0{,}5 & 0{,}5\\0{,}5 & 0{,}5 & 0{,}5\\0{,}5 & 0{,}5 & 0{,}5\end{bmatrix}\ V(1) = \begin{bmatrix}0{,}5\\0{,}5\\0{,}5\end{bmatrix}$$

Procedamos a calcular la actualización de los pesos en la matriz

$$U_{1,1}(2) = U_{1,1}(1) - \alpha\frac{\partial L}{\partial U_{1,1}}$$

donde

$$\frac{\partial L}{\partial U_{1,1}} = \sum_{t=0}^{\tau=3}\frac{\partial L^{(t)}}{\partial U_{1,1}}$$

Ya se demostró que:

$$\frac{\partial L^{(t)}}{\partial U_{m,d}} = \sum_{k=0}^{K}\left(\frac{\partial L^{(t)}}{\partial \widehat{y}_k^{(t)}}\frac{\partial \widehat{y}_k^{(t)}}{\partial o_k^{(t)}}\frac{\partial o_k^{(t)}}{\partial h_m^{(t)}}\right)\sum_{\tau=0}^{t-1}\delta_m^{u^{(t)}}\delta_m^{u^{(t-1)}}\cdots\delta_m^{u^{(\tau)}}\frac{\partial h_m^{(\tau)}}{\partial p_m^{(\tau)}}\frac{\partial p_m^{(\tau)}}{\partial U_{m,d}}$$

$$\delta_m^{u^{(t)}} = \frac{\partial h_m^{(t)}}{\partial h_m^{(t-1)}} = \frac{\partial}{\partial p_m^{(t)}}f^h\left(p_m^{(t)}\right)W_{m,m}$$

Donde en este caso particular:

$$\frac{\partial L^{(t)}}{\partial \widehat{y}^{(t)}} = \left(\widehat{y}^{(t)} - y^{(t)}\right)$$

$$\frac{\partial \widehat{y}^{(t)}}{\partial o^{(t)}} = 1$$
$$\frac{\partial o^{(t)}}{\partial h_m^{(t)}} = V_{1,m}$$
$$\delta_m^{u^{(t)}} = W_{m,m}$$
$$\frac{\partial h_m^{(\tau)}}{\partial p_m^{(\tau)}} = 1$$
$$\frac{\partial p_m^{(\tau)}}{\partial U_{m,d}} = x^{(\tau)}$$

Por lo que entonces:

$$\frac{\partial L^{(t)}}{\partial U_{m,d}} = \left(\left(\widehat{y}^{(t)} - y^{(t)}\right) V_{1,m}\right) \sum_{\tau=0}^{t-1} \left(W_{m,m}\right)^{\tau+1} x^{(\tau)}$$

con lo que para $\frac{\partial L^{(3)}}{\partial U_{1,1}}$

$$\frac{\partial L^{(3)}}{\partial U_{1,1}} = \left(\left(\widehat{y}^{(3)} - y^{(3)}\right) V_{1,1}\right) \sum_{\tau=0}^{t-1} \left(W_{1,1}\right)^{\tau+1} x^{(\tau)}$$

$$\Rightarrow \frac{\partial L^{(3)}}{\partial U_{1,1}} = \left(\left(\widehat{y}^{(3)} - y^{(3)}\right) V_{1,1}\right) \left(W_{1,1}x^{(0)} + \left(W_{1,1}\right)^2 x^{(1)} + \left(W_{1,1}\right)^3 x^{(1)}\right)$$

Observe que $W_{1,1} < 1$, por lo que el valor resultante será un valor muy pequeño. En general, conforme la secuencia es más grande y t crece, **realizaremos muchas multiplicaciones sucesivas por el factor** $W_{1,1}$, de ya sea pesos pequeños o grandes, lo que conducel al problema del **desvanecimiento del gradiente** o **explosión del gradiente**. Este problema es profundizado en el artículo *Learning long-term dependencies with gradient descent is dificult.* Una forma de lidiar con el problema de lidiar con la explosión del gradiente por ejemplo, es escalar el gradiente en caso de exceder un umbral ρ, lo cual se conoce como **saturación del gradiente o *gradient clipping***. Sin embargo para el problema del desvanecimiento del gradiente, nuevas arquitecturas para una unidad de la red recurrente.

1.4 Memorias de corto-largo plazo (LSTMs)

Las memorias de corto-largo plazo proponen computar un "olvido" de los estados de celdas anteriores, para evitar las multiplicaciones sucesivas de los parámetros W.

Cada unidad o casilla t de una red recurrente implementa las siguientes ecuaciones con $V \in \mathbb{R}^{K\times M}$,$W \in \mathbb{R}^{M\times M}$ y $U \in \mathbb{R}^{M\times D}$, $\overrightarrow{h}^{(t)} \in \mathbb{R}^M$, $\overrightarrow{x}^{(t)} \in \mathbb{R}^D$:

$$\overrightarrow{h}^{(t)} = f^h\left(\overrightarrow{p}^{(t)}\right) = f^h\left(W\overrightarrow{h}^{(t-1)} + U\overrightarrow{x}^{(t)}\right) \tag{9}$$

$$\overrightarrow{o}^{(t)} = V\overrightarrow{h}^{(t)}$$
$$\overrightarrow{\hat{y}}^{(t)} = f^o\left(\overrightarrow{o}^{(t)}\right)$$

Las memorias de corto plazo proponen crear un **estado** $\overrightarrow{s}^{(t)} \in \mathbb{R}^M$ dentro de cada unidad recurrente, el cual permita olvidar selectivamente estados de las unidades anteriores al computar la salida de las neuronas ocultas $\overrightarrow{h}^{(t)}$ (modificando entonces la Ecuación 9), implementando la siguiente ecuación de actualización de estado:

$$\overrightarrow{s}^{(t)} = \overrightarrow{f}^{(t)} . \cdot \overrightarrow{s}^{(t-1)} + \overrightarrow{g}^{(t)} . \cdot \text{sigmoid}\left(W\overrightarrow{h}^{(t-1)} + U\overrightarrow{x}^{(t)}\right)$$

Donde:

1. **Compuertas de olvido** $\overrightarrow{f}^{(t)} \in \mathbb{R}^M$: cada componente $0 \leq f_i^{(t)} \leq 1$ define por cuanto el componente i del estado $\overrightarrow{s}^{(t-1)}$ de "olvidarse" , al realizarse la multiplicación por elemento correspondiente (un valor de salida 0 indica que se "olvide del todo" del estado anterior de la celda, y una salida 1 que lo recuerde). Ese "olvido" se computa, usango una función de activación sigmoidal que garantice que los valores estarán entre 0 y 1, **para cada casilla** t:

   $$\overrightarrow{f}^{(t)} = \text{sigmoid}\left(W^f\overrightarrow{h}^{(t-1)} + U^f\overrightarrow{x}^{(t)}\right)$$

   donde se definen matrices de parámetros específicas para computar el "olvido", $W^f \in \mathbb{R}^{M\times M}$ y $U^f \in \mathbb{R}^{M\times D}$.

2. **Compuertas de entradas externas** $\overrightarrow{g}^{(t)} \in \mathbb{R}^M$: Estas compuertas controlan la influencia de las unidades ocultas anteriores combinadas linealmente por W, U, por cada componente, de modo que e $0 \leq g_i^{(t)} \leq 1$ al estado actual $\overrightarrow{s}^{(t)}$

   $$\overrightarrow{g}^{(t)} = \text{sigmoid}\left(W^g\overrightarrow{h}^{(t-1)} + U^g\overrightarrow{x}^{(t)}\right).$$

   O en otras palabras controla que **componentes del arreglo** $\overrightarrow{i}^{(t)}$**se escriben en el estado de la** celda. El control se ejerce a través de sus matrices de parámetros específicos $W^g \in \mathbb{R}^{M\times M}$ y $U^g \in \mathbb{R}^{M\times D}$ para computar la contribución de las unidades ocultas de la casilla anterior $t-1$, modulando la señal de las unidades ocultas y la muestra a través de la multiplicación por elemento, en lo que se agrupa usualmente como la **señal de entrada** o *input*:

   $$\overrightarrow{i}^{(t)} = \text{sigmoid}\left(W\overrightarrow{h}^{(t-1)} + U\overrightarrow{x}^{(t)}\right)$$

La salida del estado de la celda $\overrightarrow{s}^{(t)}$ no debe quedar directamente expuesta a las demás casillas $t-k$ anteriores para evitar el desvanecimiento del gradiente,

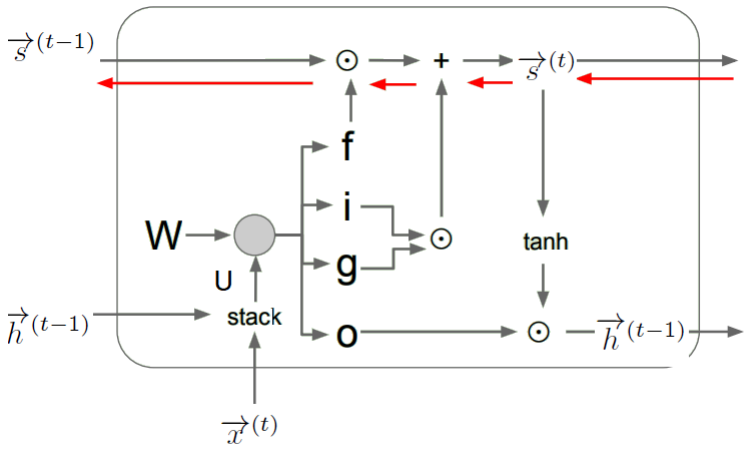


Figura 15: Flujo de los gradientes en una celda LSTM.

por lo que de nuevo, se emplea el enfoque de multiplicación por elemento para activar y desactivar el aporte del estado de la celda a la salida de las neuronas:

$$\overrightarrow{h}^{(t)} = \tanh\left(\overrightarrow{s}^{(t)}\right) . \cdot \overrightarrow{q}^{(t)}$$

donde de forma similar se definen las **compuertas de salida** u *output gates:*

$$\overrightarrow{q}^{(t)} = \text{sigmoid}\left(W^q \overrightarrow{h}^{(t-1)} + U^q \overrightarrow{x}^{(t)}\right).$$

Con parámetros específicos $W^q \in \mathbb{R}^{M \times M}$ y $U^q \in \mathbb{R}^{M \times D}$. Las implementaciones más simples hacen que $W = W^q = W^g = W^f$ y $U = U^q = U^g = U^f$

La propagación hacia atrás en $\overrightarrow{s}^{(t)}$ a $\overrightarrow{s}^{(t-1)}$ se realiza entonces con una multiplicación por elemento de $\overrightarrow{f}^{(t)}$ y no por la multiplicación de W, por lo que el gradiente fluye en la dirección demarcada en la Figura 15 cuando las compuertas de olvido están activadas.

2 Redes recurrentes profundas

Existen tres bloques básicos en las redes profundas, el bloque de conexión de la entrada al estado oculto, de estado oculto a estado oculto y del estado oculto a la salida. Tales tres variantes pueden ser profundizadas, individualmente y en conjunto, y artículos como (*Graves et al., 2013; Pascanu et al., 2014a*) sugieren que profundizando tales bloques genera mejores resultados. La Figura 16 muestra tales variantes:

La Figura 17 muestra la variante (a) de la Figura 16, *desenvuelta* y con tres capas. Por defecto las **capas se asumen completamente conectadas**, pero es

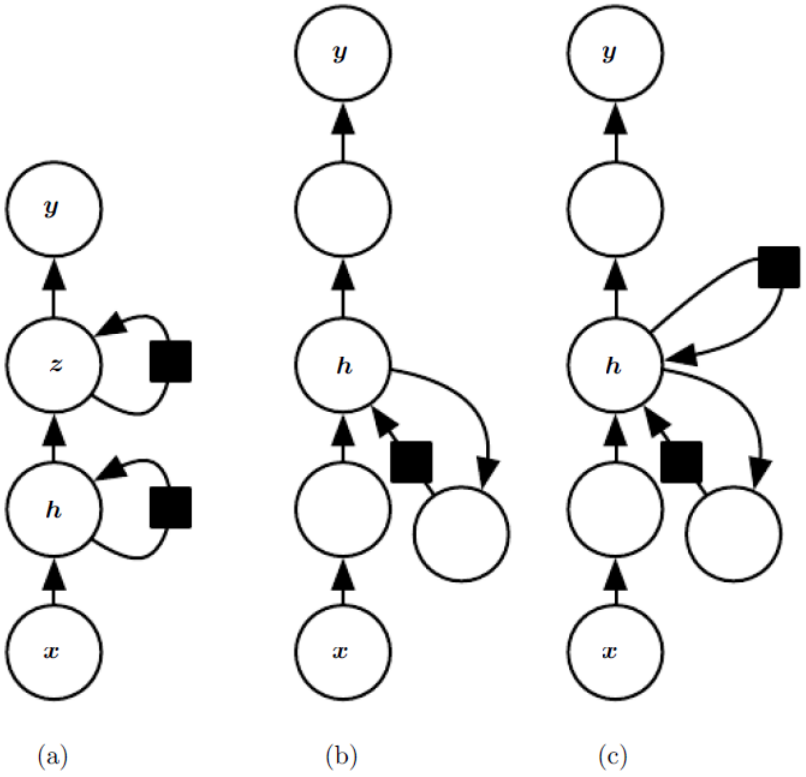


Figura 16: Variantes de profundización de una red recurrente, tomado de [1].

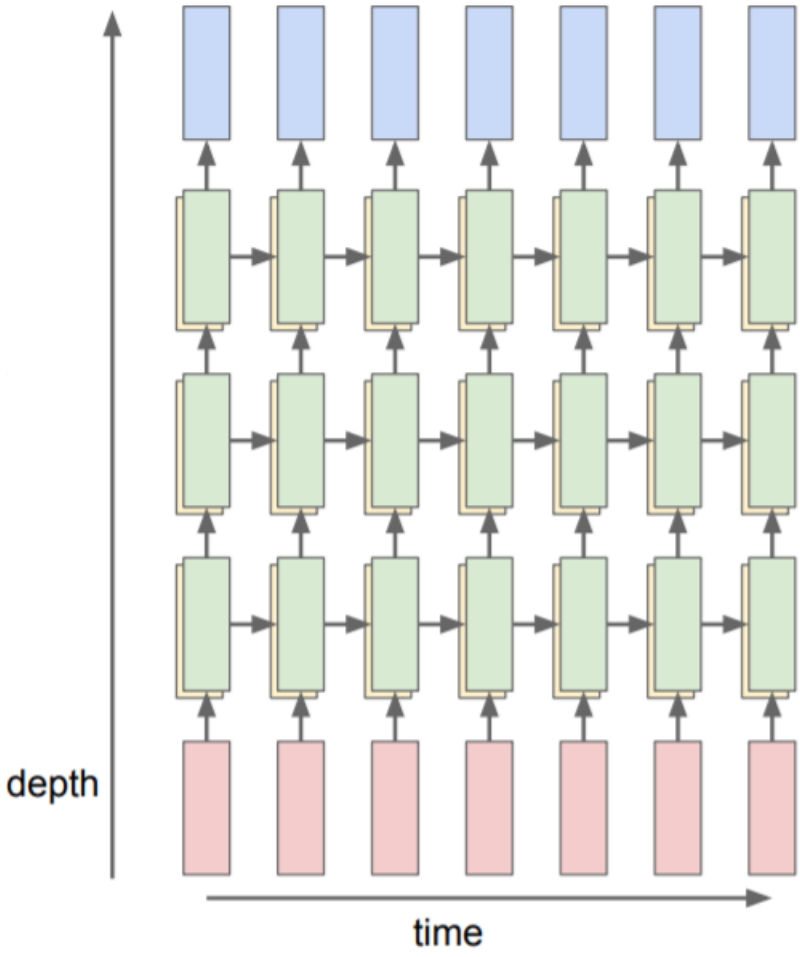


Figura 17: Red recurrente con tres capas ocultas completamente conectadas.

usual utilizar **esquemas de conexión local como las redes convolucionales**. No es usual utilizar redes con más de 4 capas.

2.1 Red recurrente de Elman para estimación del valor en *t* para series temporales continuas

La red de Elman publicada en el artículo *Finding structure in time* define las ***neuronas o unidades de contexto*** las cuales reciben su entrada de las unidades ocultas en la casilla o instante anterior $t-1$, por lo que se define una neurona de contexto por cada neurona en la capa oculta. La matriz de pesos $W \in \mathbb{R}^{M \times M}$ conecta las neuronas de contexto $\overrightarrow{h}^{(t-1)} = \overrightarrow{c}^{(t)} \in \mathbb{R}^M$ con las neuronas de la capa oculta $\overrightarrow{h}^{(t)} \in \mathbb{R}^M$, mientras que la matriz $U \in \mathbb{R}^{D \times M}$ conecta la entrada

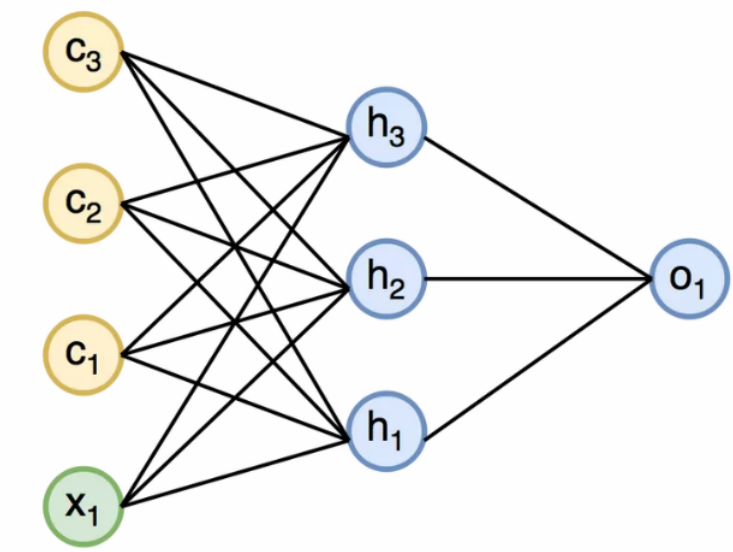


Figura 18: Red de Elmann para la predicción de una función.

$\overrightarrow{x}^{(t)} \in \mathbb{R}^D$ con las unidades en la capa oculta $\overrightarrow{h}^{(t)}$, según la definición realizada anteriormente de las redes recurrentes. La red de Elman define una matriz W_1 la cual contenga los pesos definidos en las matrices U y W, de forma que al arreglo de entrada se le concatenan las unidades de contexto, por lo que entonces $W_1 \in \mathbb{R}^{M \times (M+D)}$. La Figura 18 muestra el ejemplo de una red de Elmann para estimar la salida de una función $f : \mathbb{R} \to \mathbb{R}$, donde entonces $D = 1$, $M = 3$ y $K = 1$. La red en este caso se implementa para realizar la estimación de un valor continuo, por lo que a la salida, la función de activación corresponde a la función identidad:

$$\overrightarrow{y}^{(t)} = g^o\left(\overrightarrow{o}^{(t)}\right) = \overrightarrow{o}^{(t)} = V\overrightarrow{h}^{(t)}$$

y para la capa oculta, se utiliza una función acotadora como el tangente hiperbólico:

$$\overrightarrow{h}^{(t)} = \tanh\left(\overrightarrow{p}^{(t)}\right) = \tanh\left(W\overrightarrow{h}^{(t-1)} + U\overrightarrow{x}^{(t)}\right) = \tanh\left(W_1\left[\overrightarrow{c}^{(t)}\ \overrightarrow{x}^{(t)}\right]\right)$$

donde $\left[\overrightarrow{c}^{(t)}\ \overrightarrow{x}^{(t)}\right]$ corresponde a la concatenación del vector de entrada y la salida de las neuronas de contexto.

3 Red recurrente con un solo vector $\overrightarrow{x}$ como entrada

Tómese el problema de generar o estimar una secuencia de valores $\overrightarrow{y}^{(1)}, \overrightarrow{y}^{(2)}, \ldots, \overrightarrow{y}^{(\tau)}$ a partir de un único vector de entrada $\overrightarrow{x}$, como por ejemplo, en la aplicación de generar una descripción textual (en este caso la secuencia a estimar

$\overrightarrow{\hat{y}}^{(1)}, \overrightarrow{\hat{y}}^{(2)}, \ldots, \overrightarrow{\hat{y}}^{(\tau)}$ corresponde a una secuencia de palabras) a partir de una imagen $\overrightarrow{x}$. Para ello se define la matriz de parámetros R la cual asocia o mapea la entrada $\overrightarrow{x}$, y una matriz U que asocia las salidas anteriores con el estado actual.

Referencias

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

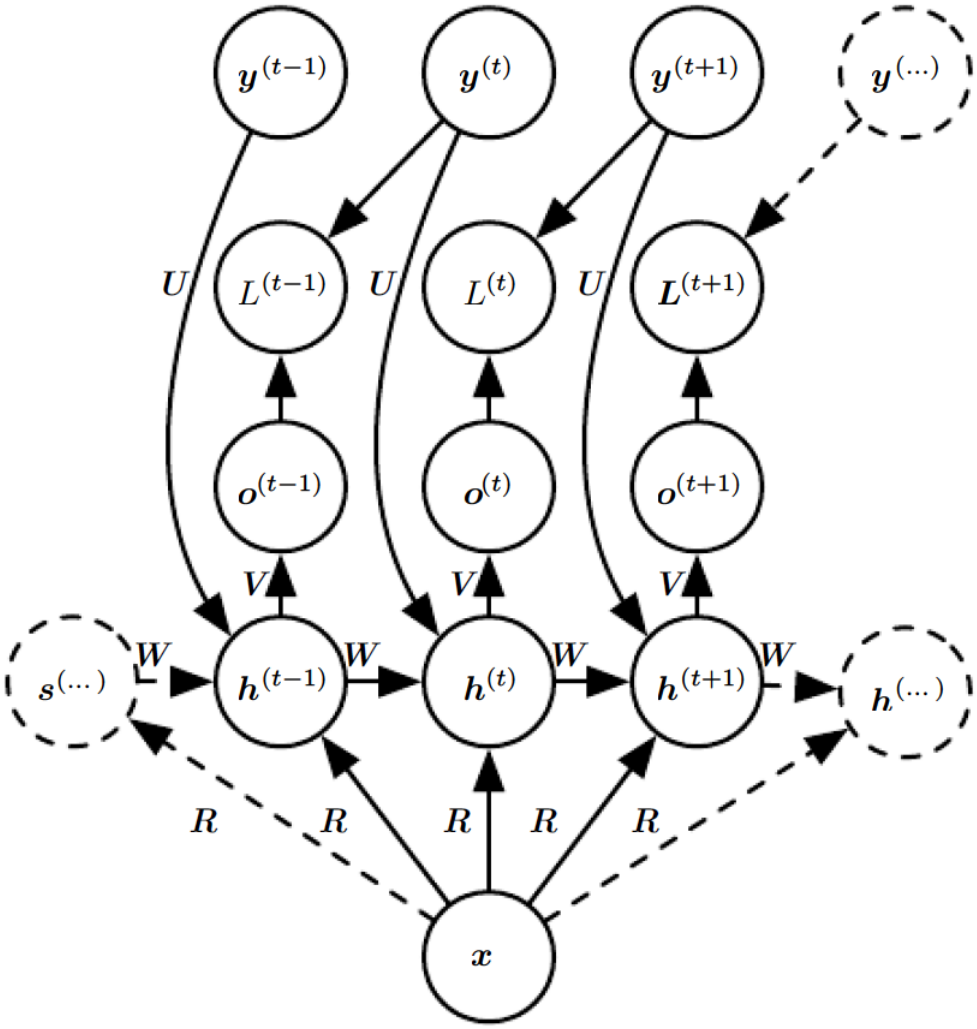


Figura 19: Red de mapeo o predicción de una secuencia desde una imagen.