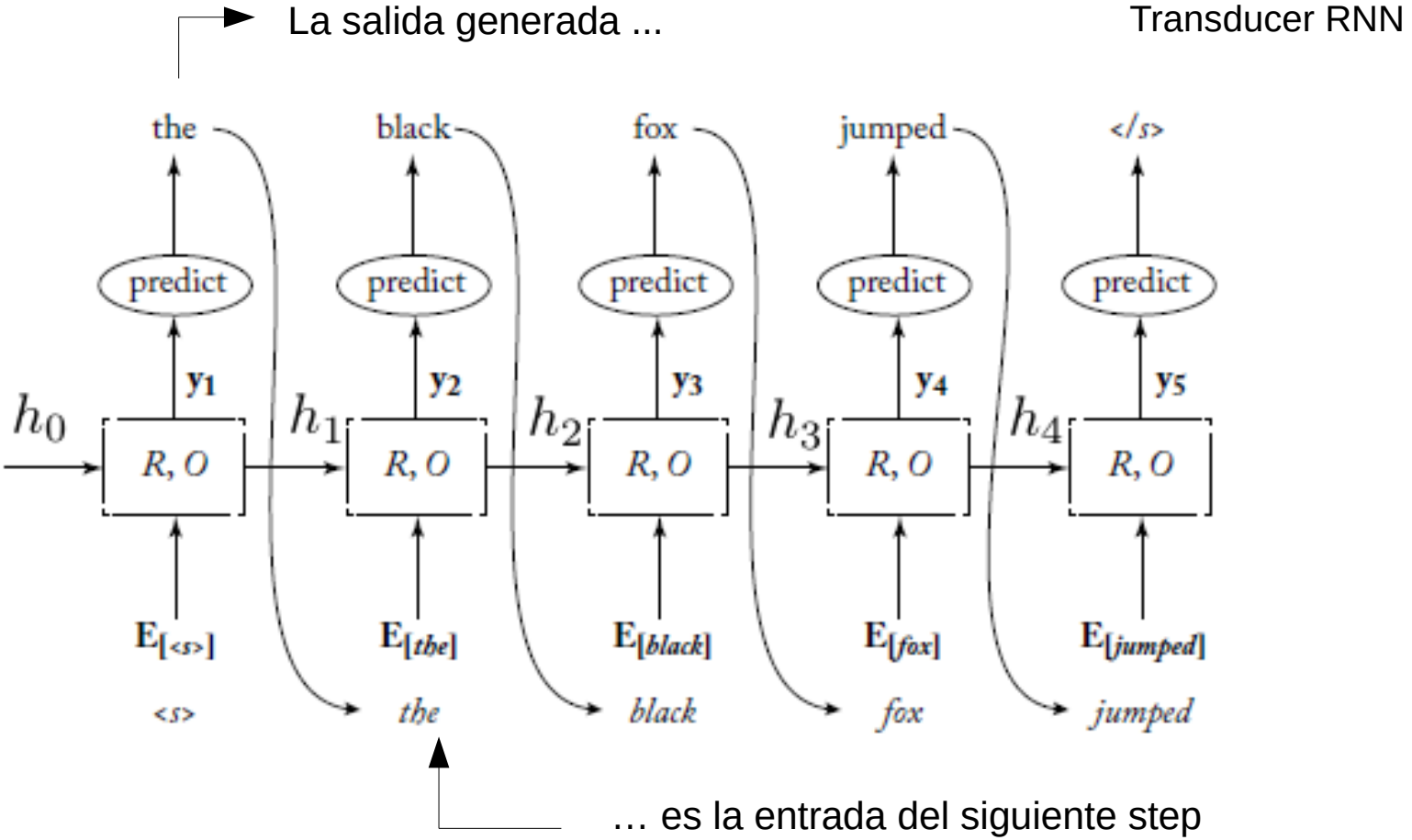
IIC3670 Procesamiento de Lenguaje Natural

https://github.com/marcelomendoza/IIC3670

- GENERACIÓN DE TEXTO -

Generación de texto

Generación auto-regresiva



Generación de texto

Generación condicional

$$\hat{t}_{j+1} \sim p(t_{j+1} = k \mid \hat{t}_{1:j}).$$

Transducer RNN:

$$p(t_{j+1} = k \mid \hat{t}_{1:j}) = f(\mathrm{RNN}(\hat{\boldsymbol{t}}_{\mathbf{1:j}}))$$

$$p(t_{j+1} = k|\hat{t}_{1:j}) = f(O(h_{j+1}))$$

Generación de texto

Generación condicional

$$\hat{t}_{j+1} \sim p(t_{j+1} = k \mid \hat{t}_{1:j}).$$

Transducer RNN:

$$p(t_{j+1} = k \mid \hat{t}_{1:j}) = f(\mathrm{RNN}(\hat{\boldsymbol{t}}_{\mathbf{1:j}}))$$

$$p(t_{j+1} = k|\hat{t}_{1:j}) = f(O(h_{j+1}))$$

vector de contexto

Transducer RNN condicional:

$$p(t_{j+1} = k \mid \hat{t}_{1:j}, c) = f(\mathrm{RNN}(\boldsymbol{v}_{\mathbf{1:j}}))$$

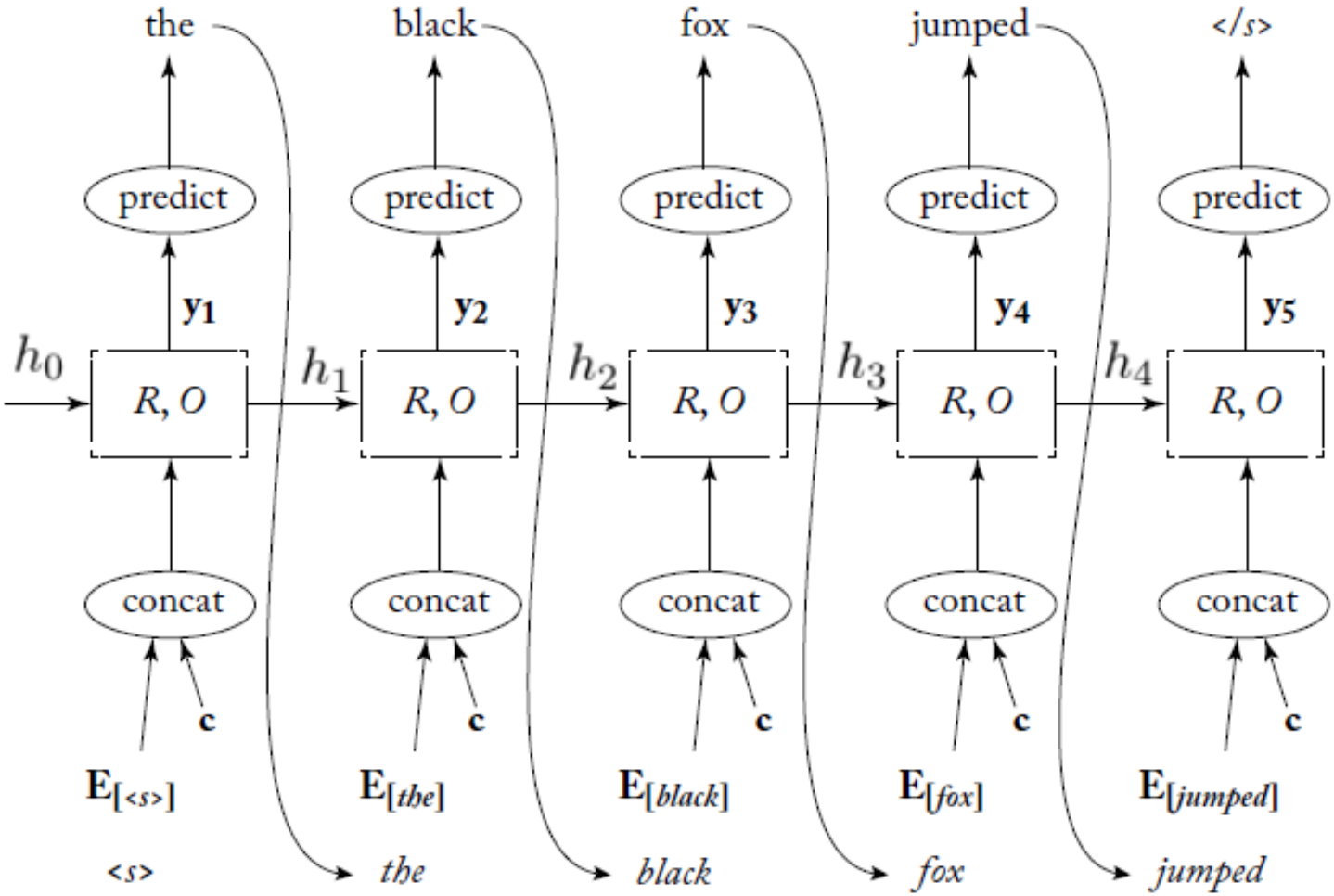
$$\boldsymbol{v}_i = [\hat{\boldsymbol{t}}_i; c]$$

$$\hat{t}_j \sim p(t_j \mid \hat{t}_{1:j-1}, c),$$

Generación de texto

c puede representar un tema u otra sentencia

Transducer RNN condicional (generador):



Generación de texto

Generación condicional (encoder-decoder)

Sequence to sequence: **c** es un vector que representa una secuencia de texto de entrada (encoder)

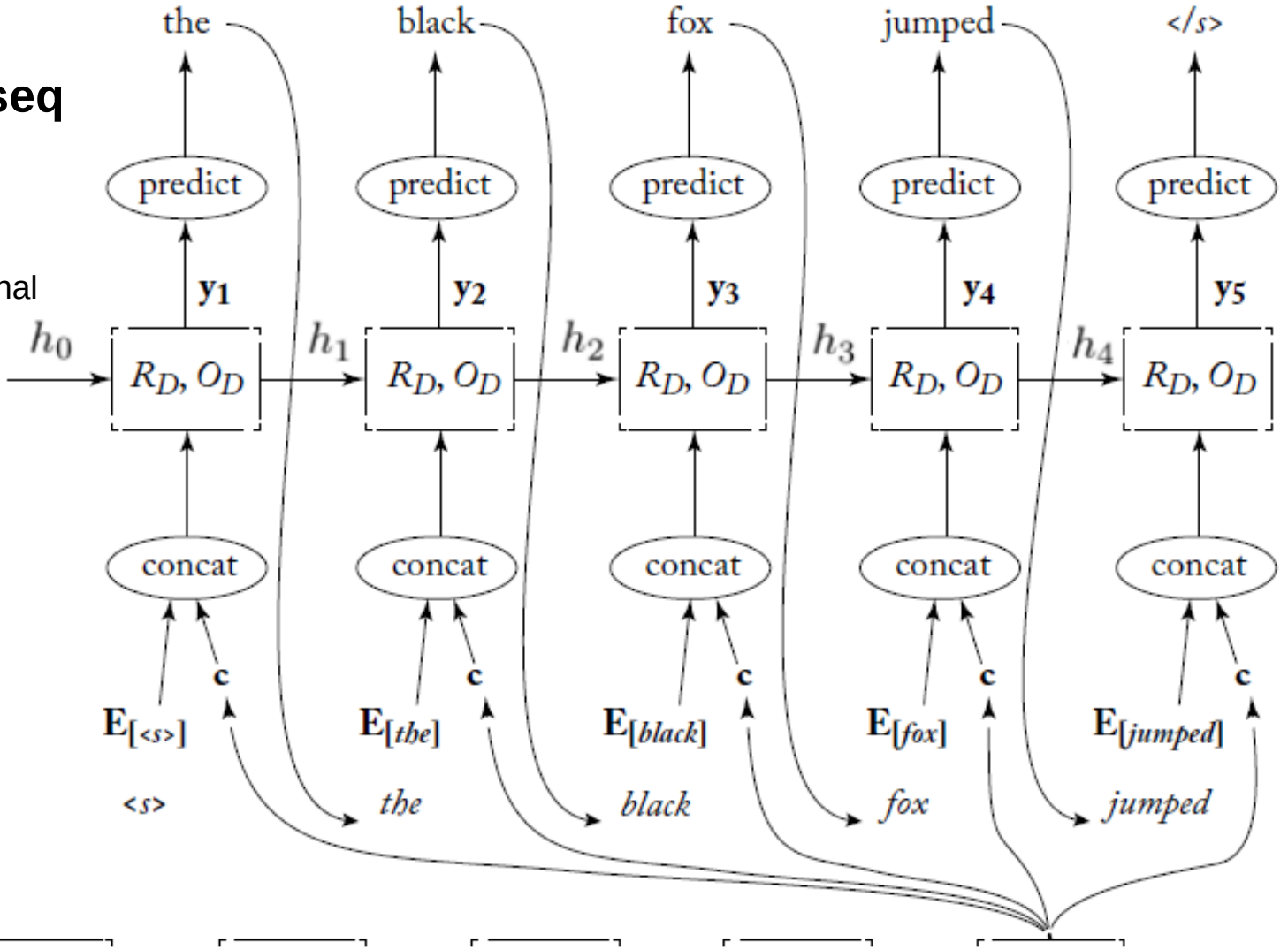
source sequence $x_{1:n}$ → target output $t_{1:m}$

Encoder: $\text{RNN: } c = \text{RNN}^{\text{enc}}(x_{1:n}).$

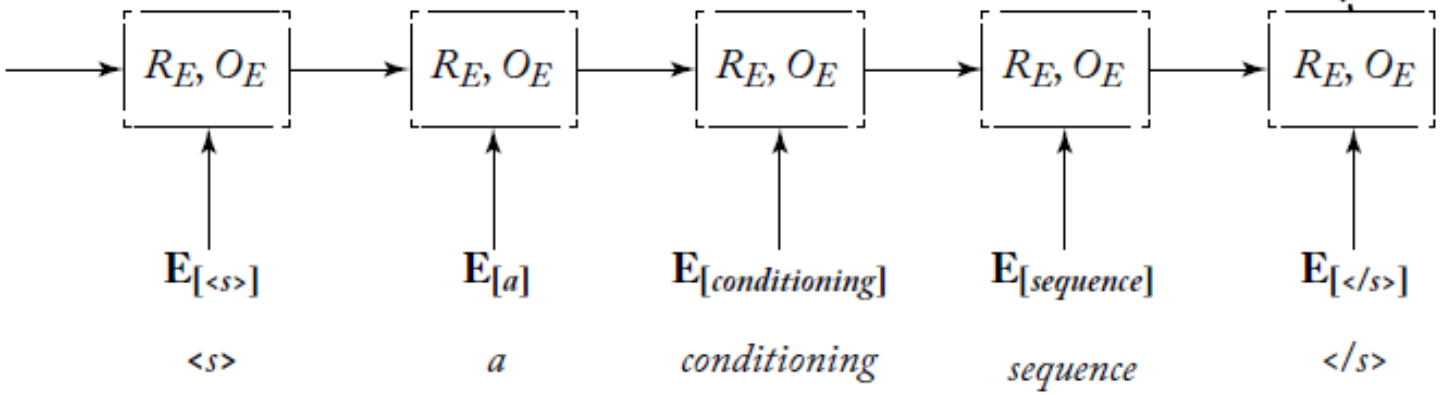
Decoder:
$$p(t_{j+1} = k \mid \hat{t}_{1:j}, c) = f(\text{RNN}(v_{1:j}))$$
$$v_i = [\hat{t}_i; c]$$
$$\hat{t}_j \sim p(t_j \mid \hat{t}_{1:j-1}, c),$$

Seq2seq

Transducer RNN condicional



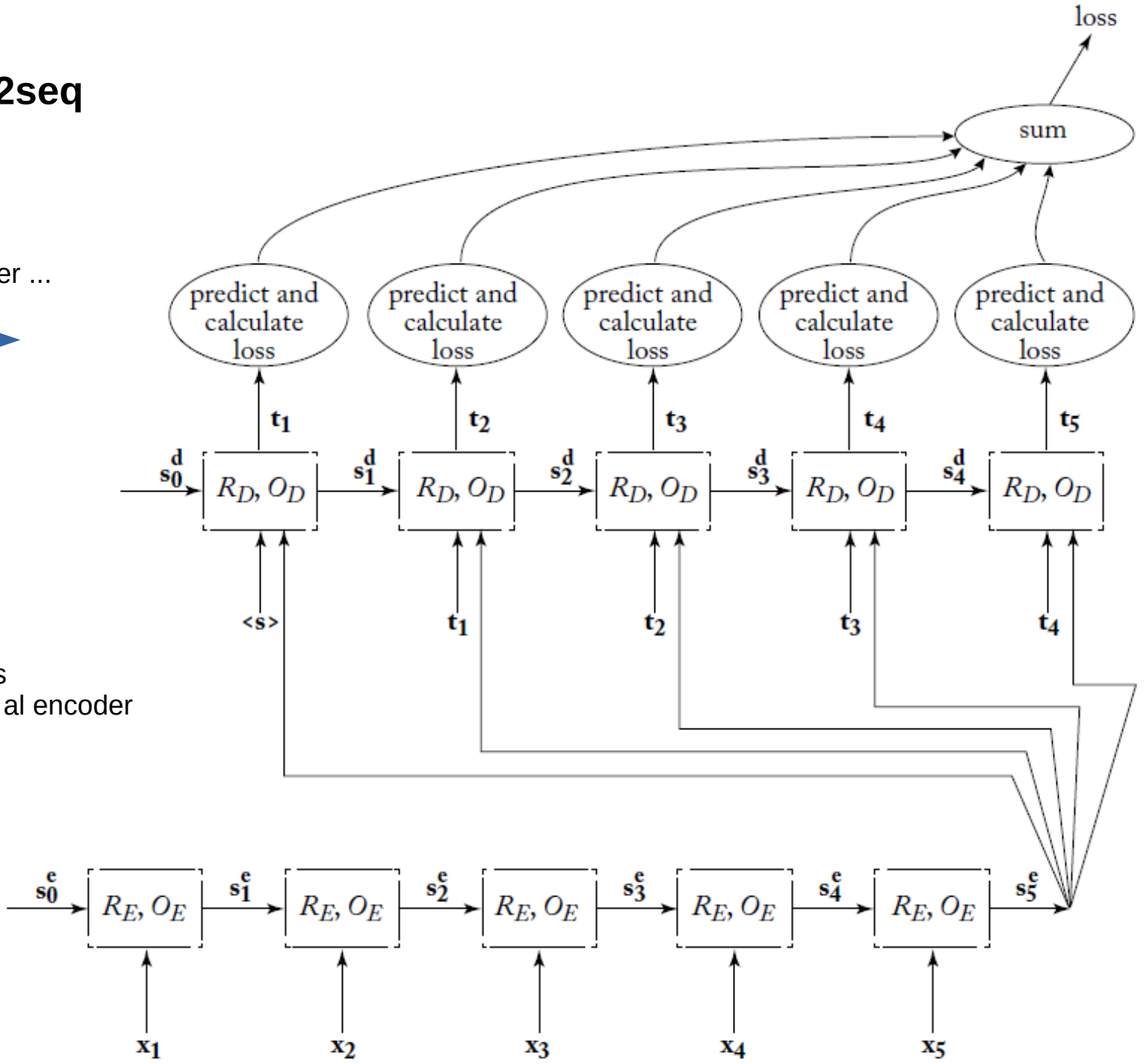
RNN Encoder



Seq2seq

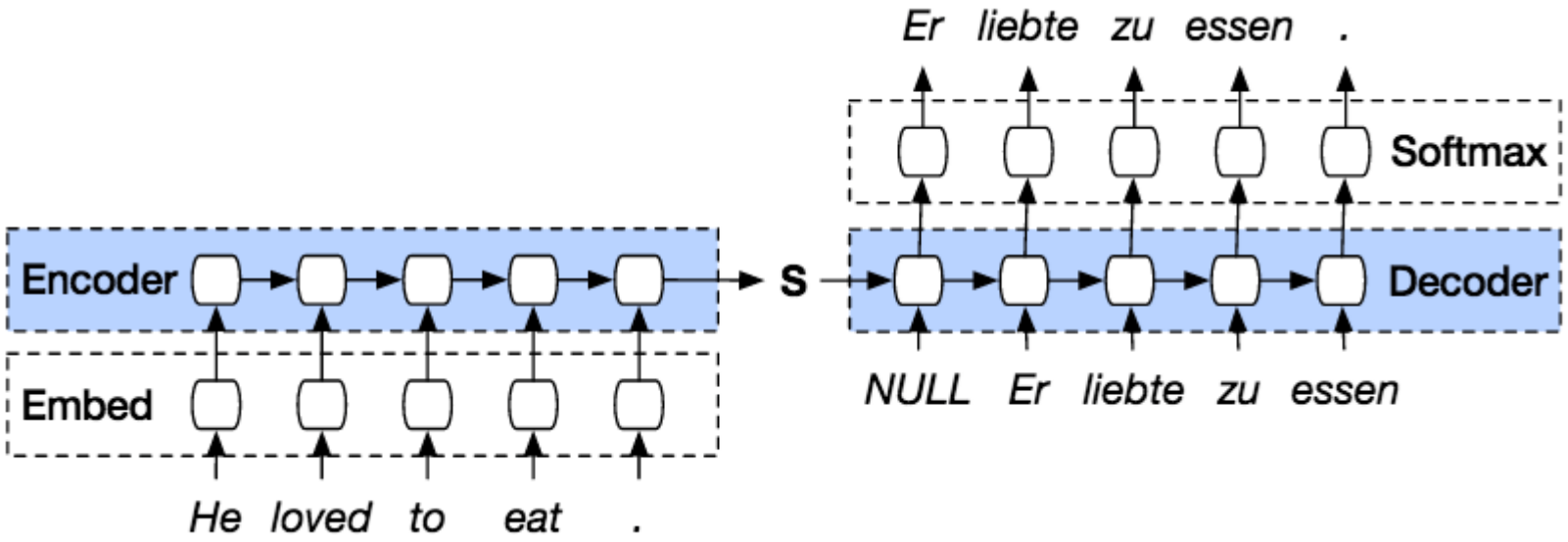
La supervisión se hace en el decoder ...

… pero los gradientes también se propagan al encoder



Seq2seq

- Aplicación más conocida de la arquitectura: **Machine translation**

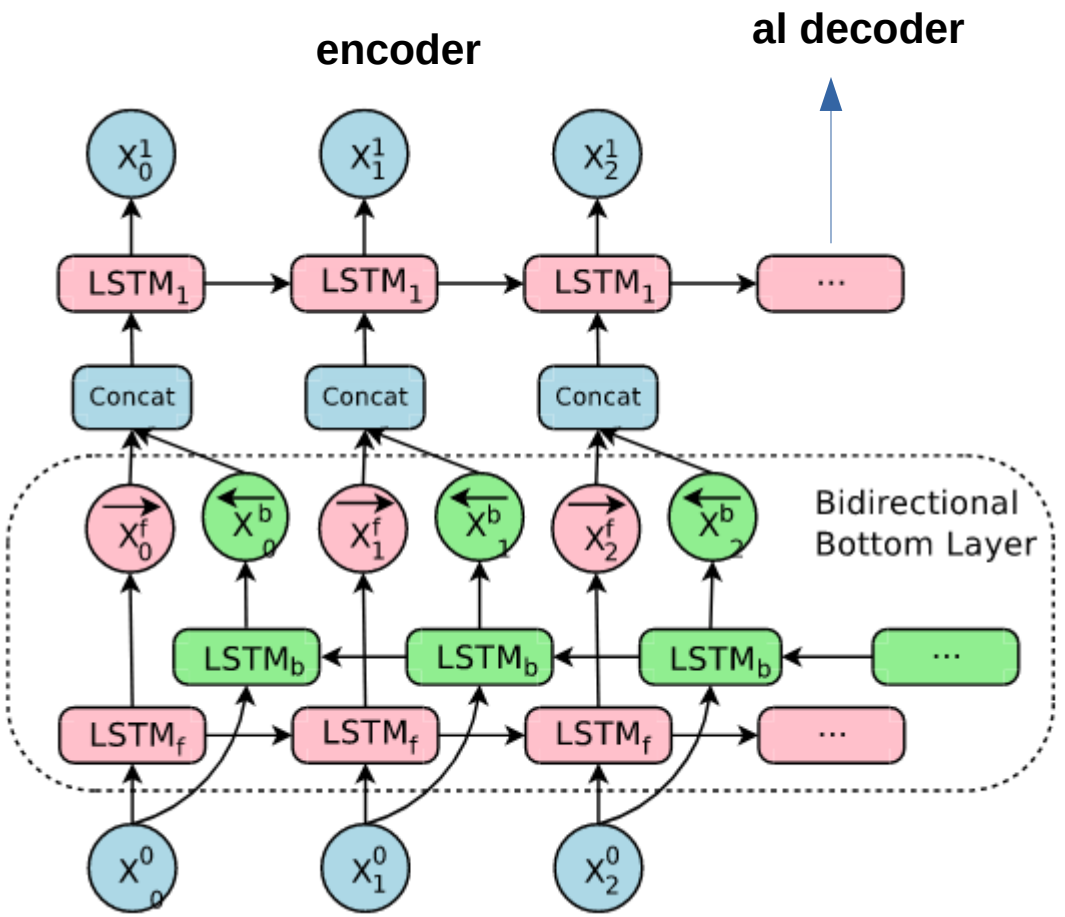


- Otras: **QA (closed)**

Seq2seq

Google Machine translation:

Usan un **encoder** más complejo



Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation, https://arxiv.org/pdf/1609.08144.pdf , 2016

- GENERACIÓN CONDICIONAL CON ATENCIÓN -

Generación condicional con atención

Idea: usar un mecanismo de atención en el **decoder**

Largo de la sentencia de entrada (encoder)

Posición en la sentencia de salida (decoder)

$$\text{attend}(c_{1:n}, \hat{t}_{1:j}) = c^j$$

factores de atención

vectores de contexto (encoder)

$$c^j = \sum_{i=1}^{n} \alpha^j_{[i]} \cdot c_i$$

los factores se aprenden con una **softmax**

$$\alpha^j = \text{softmax}(\bar{\alpha}^j_{[1]}, \ldots, \bar{\alpha}^j_{[n]})$$

$$\bar{\alpha}^j_{[i]} = \text{MLP}^{\text{att}}([h_j ; c_i])$$

la **softmax** opera a la salida de una MLP que opera sobre h_j y c_i

Ojo, h del decoder!!!

Generación condicional con atención

encoder

$$c_{1:n} = \text{biRNN}^{\star}_{\text{enc}}(x_{1:n})$$

decoder

$$\bar{\alpha}^{j}_{[i]} = \text{MLP}^{\text{att}}(h_j; c_i) \rightarrow \text{MLP}^{\text{att}}([h_j; c_i]) = v \tanh([h_j; c_i]U + b)$$

encoder

soft atenttion

$$\alpha^{j} = \text{softmax}(\bar{\alpha}^{j}_{[1]}, \ldots, \bar{\alpha}^{j}_{[n]})$$

$$c^{j} = \sum_{i=1}^{n} \alpha^{j}_{[i]} \cdot c_i$$

$$p(t_{j+1} = k \mid \hat{t}_{1:j}, x_{1:n}) = f(O_{\text{dec}}(h_{j+1}))$$

$$h_{j+1} = R_{\text{dec}}(h_j, [\hat{t}_j; c^{j}])$$

decoder

Generación condicional con atención

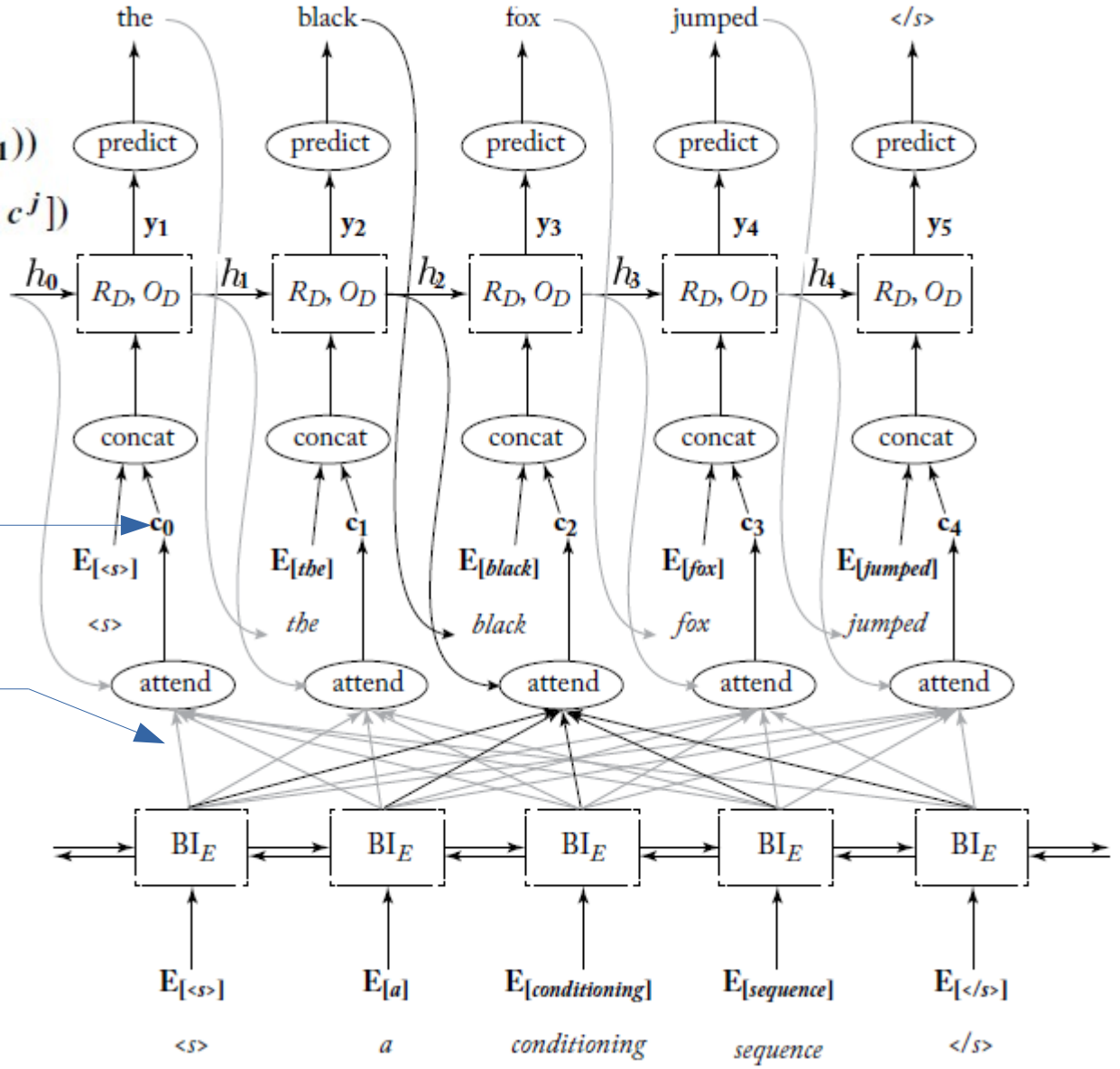
$$p(t_{j+1} = k \mid \hat{t}_{1:j}, x_{1:n}) = f(O_{\text{dec}}(h_{j+1}))$$

$$h_{j+1} = R_{\text{dec}}(h_j, [\hat{t}_j; c^j])$$

$$c^j = \sum_{i=1}^{n} \alpha^j_{[i]} \cdot c_i$$

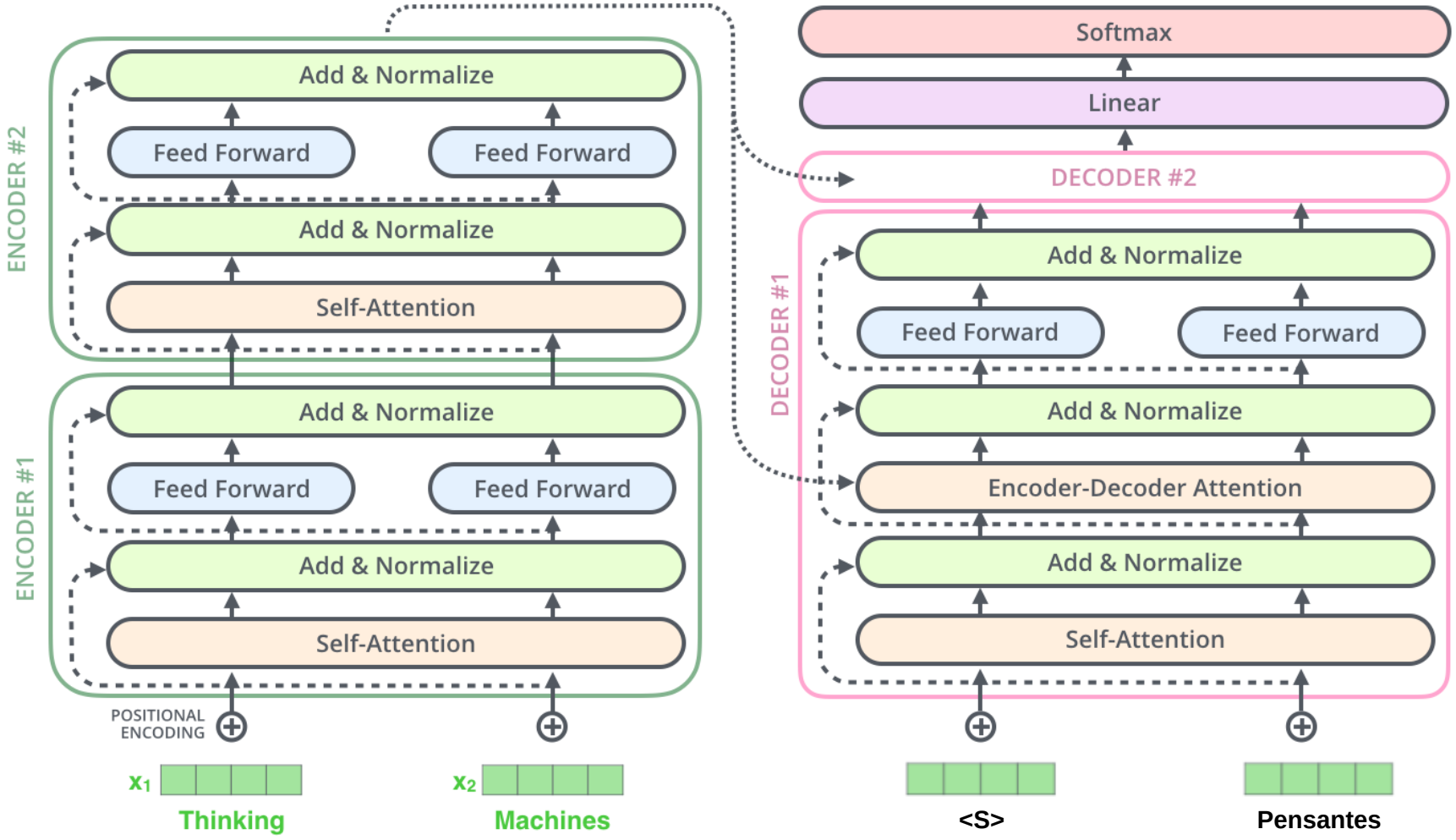
$$\bar{\alpha}^j_{[i]} = \text{MLP}^{\text{att}}(h_j; c_i])$$

$$c_{1:n} = \text{biRNN}^{\star}_{\text{enc}}(x_{1:n})$$

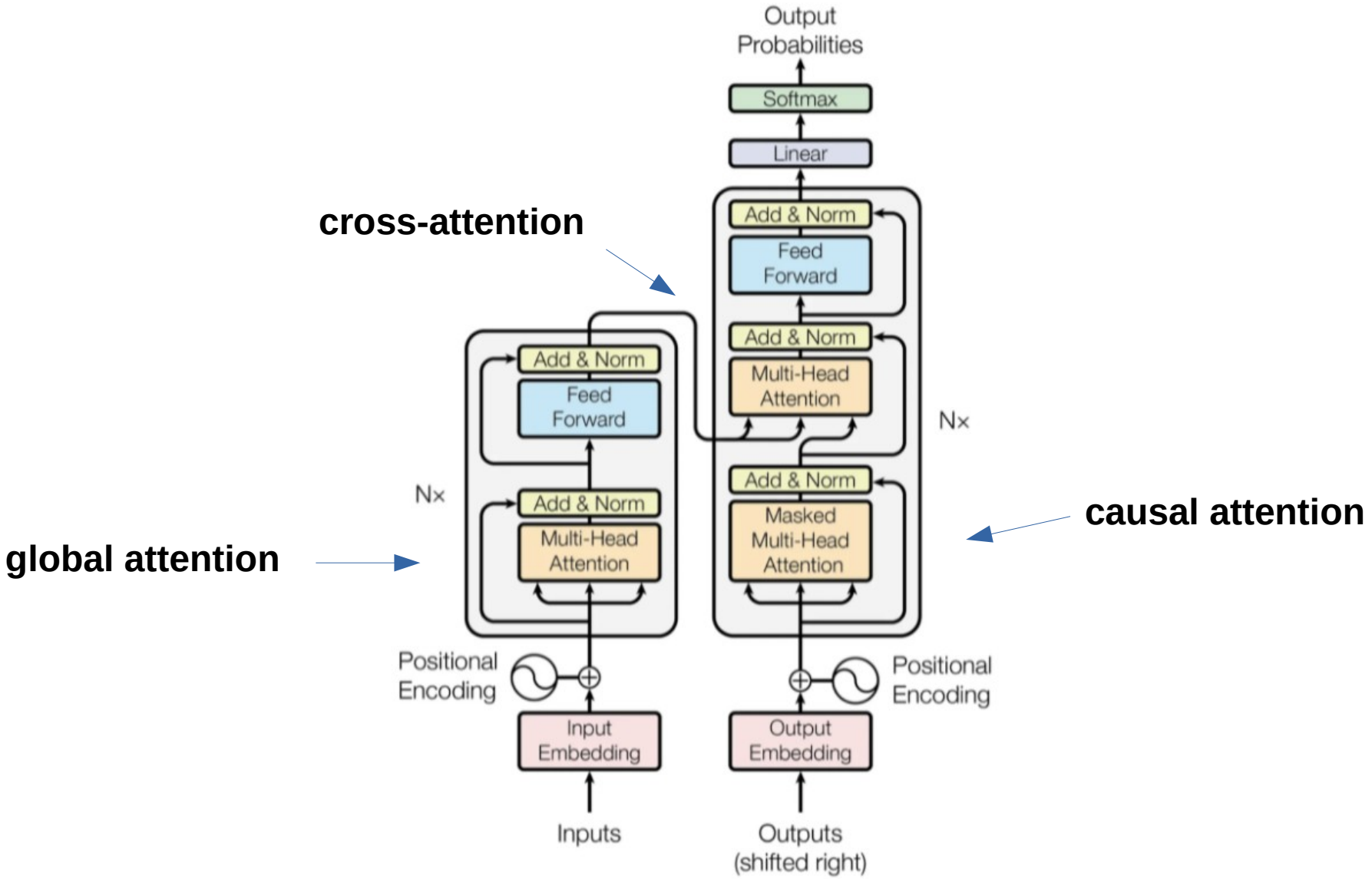


- GENERACIÓN CONDICIONAL CON TRANSFORMERS -

Generación condicional con atención (transformer seq2seq)



Generación condicional con atención (transformer seq2seq)



Generación condicional con atención (transformer seq2seq)