

Guía Práctica 3 – Modelos neuronales de etiquetado de secuencias

Grado en Inteligencia Artificial – Curso 2024/2025

Fundamentos de Procesamiento de Lenguaje Natural



UNIVERSIDADE DA CORUÑA

Etiquetado de secuencias

Asignar a cada palabra de una oración una, y sola una, etiqueta.

El	volcán	emitió	mucha	lava	durante	la	erupción
DET	NOUN	VERB	DET	NOUN	PREP	DEP	NOUN

Es la aproximación más simple de predicción estructurada, es decir, de convertir texto no estructurado, en una representación estructurada que se pueda usar para extraer información automáticamente.

Varios problemas pueden resolverse con esta aproximación...

Etiquetado de secuencias

Ventajas:

- Rápidos. Potencialmente $O(n)$ y exactamente n acciones de etiquetación, donde n es el tamaño de la oración de entrada.
- Fáciles de entender y de implementar. No son un concepto limitado a PLN, por lo que también es fácil de explicar a no expertos. Por ejemplo también se realiza etiquetado de secuencia de ADN.

Etiquetado de secuencias

Desventajas:

- En muchas ocasiones, solo permiten extraer información superficial de la oración.
- Para extraer información y relaciones más complejas entre palabras, se pueden emplear otro tipo de paradigmas o grandes modelos de lenguaje. Veremos más sobre esto en Técnicas Avanzadas de Procesamiento de Lenguaje Natural.

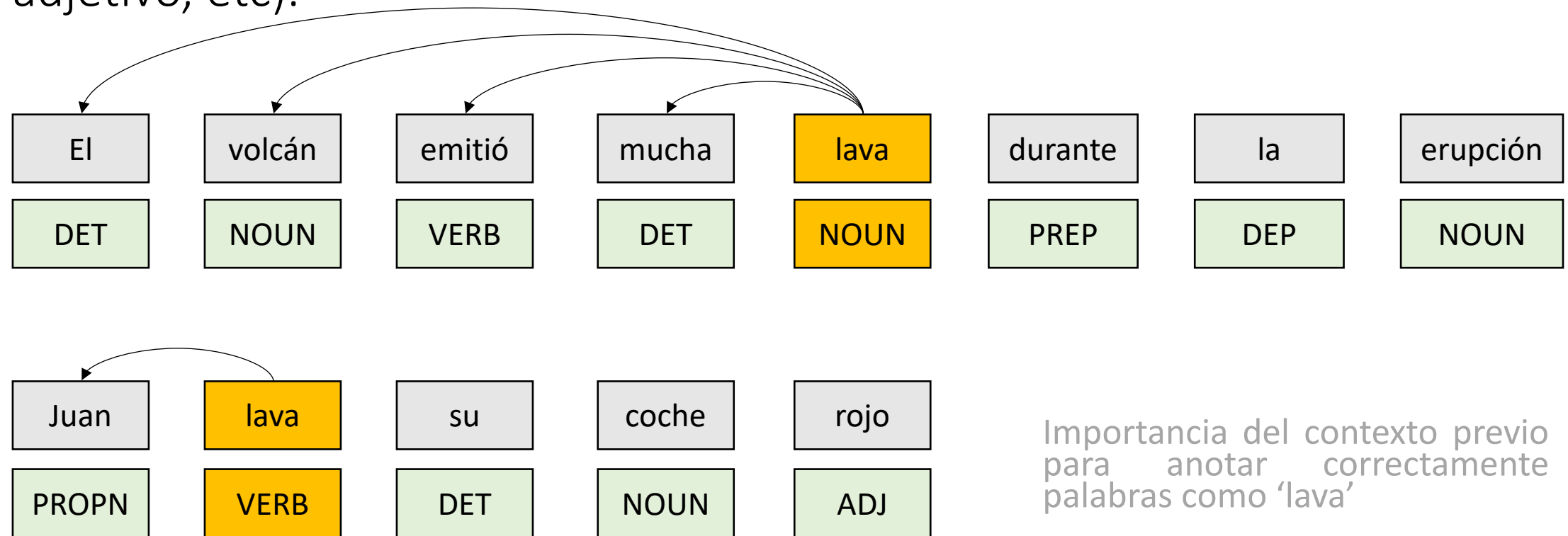
Etiquetado de secuencias

En esta práctica, vamos a ver su aplicación a dos tareas clásicas de PLN:

- Etiquetación morfológica o **part-of-speech (PoS) tagging**.
- Reconocimiento de entidades nombradas o **named-entity recognition (NER)**.

Part-of-speech tagging (PoS tagging)

Etiquetar cada palabra con su categoría gramatical (nombre, verbo, adjetivo, etc).



Named-entity recognition (NER)

Reconocimiento de entidades nombradas.

Identificar información clave en texto escrito, habiendo definido previamente un conjunto de categorías (nombres de personas, localizaciones, organizaciones, etc).

Una entidad puede definirse como un concepto del que se habla frecuentemente en un determinado dominio. Por ejemplo, nombres de actores en una película, o los platos en un menú de un restaurante.

Named-entity recognition (NER)

Juan	lava	su	coche	rojo
B-PERSON	O	O	O	O
Barack	Obama	nació	en	Hawái
B-PERSON	I-PERSON	O	O	B-LOC

Named-entity recognition (NER)

Suelen anotarse con el criterio de anotación BIO:

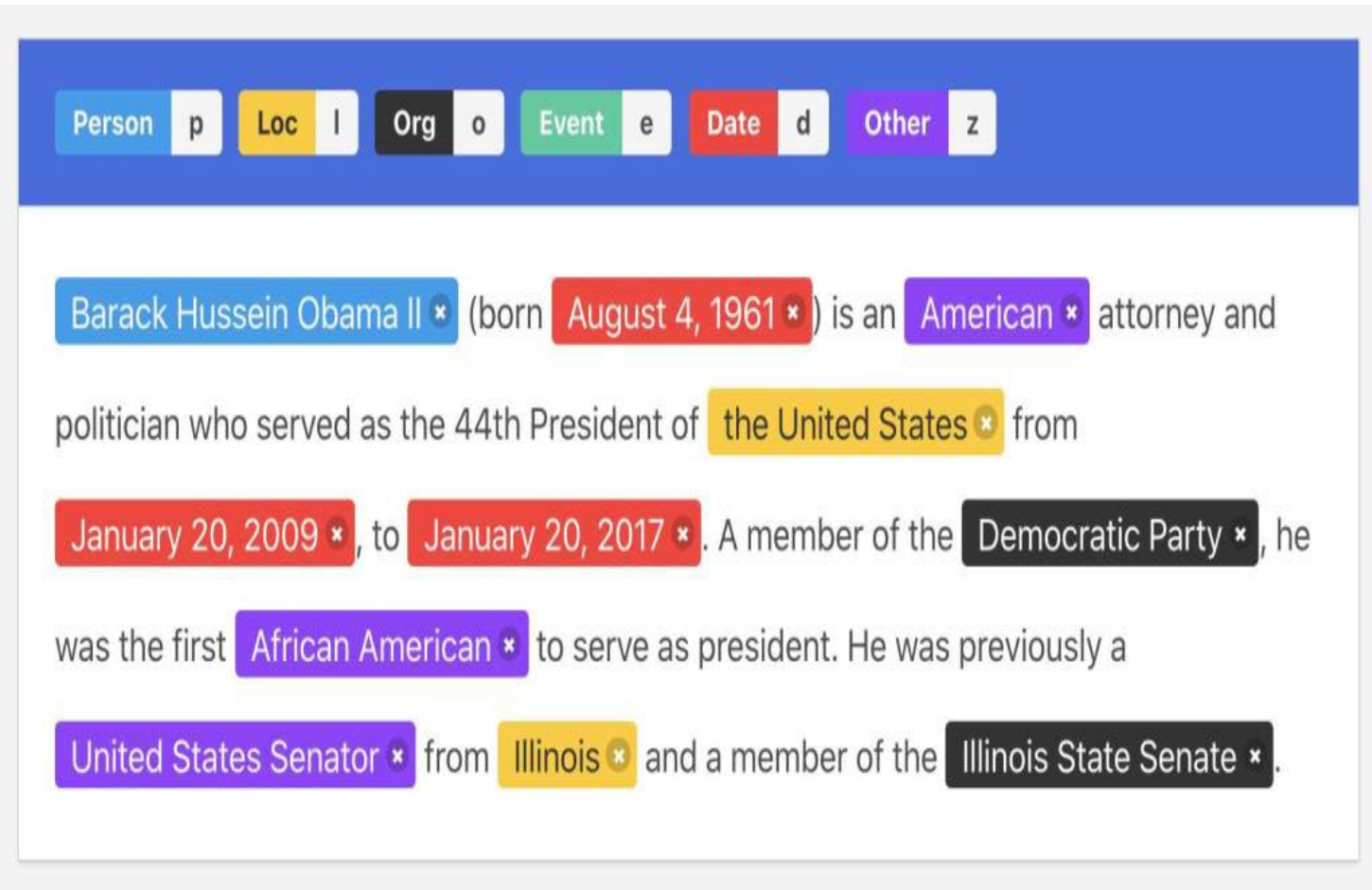
B: Indica que con esa palabra comienza una entidad.

I: Indica que con esa palabra continúa el reconocimiento de una entidad que se abrió anteriormente (etiquetando una palabra previa con una B).

O: Etiqueta usada para identificar aquellas palabras que no aportan información relevante para el dominio.

```
2    B-Rating
start  I-Rating
restaurants O
with   O
inside B-Amenity
dining I-Amenity
```

Aplicaciones de *named-entity recognition (NER)*



The screenshot displays a Named Entity Recognition (NER) interface. At the top, a blue header bar contains a legend with seven categories: Person (p, blue), Loc (l, yellow), Org (o, black), Event (e, green), Date (d, red), and Other (z, purple). Below the legend, a text snippet is shown with various entities highlighted in colored boxes and marked with an asterisk (*). The text is: "Barack Hussein Obama II * (born August 4, 1961 *) is an American * attorney and politician who served as the 44th President of the United States * from January 20, 2009 *, to January 20, 2017 *. A member of the Democratic Party *, he was the first African American * to serve as president. He was previously a United States Senator * from Illinois * and a member of the Illinois State Senate *." The entities are labeled as follows: "Barack Hussein Obama II" (Person, blue), "August 4, 1961" (Date, red), "American" (Other, purple), "the United States" (Loc, yellow), "January 20, 2009" (Date, red), "January 20, 2017" (Date, red), "Democratic Party" (Org, black), "African American" (Other, purple), "United States Senator" (Other, purple), "Illinois" (Loc, yellow), and "Illinois State Senate" (Org, black).

Extraída de: <https://www.analyticsvidhya.com/blog/2021/11/a-beginners-introduction-to-ner-named-entity-recognition/>

Aplicaciones de *named-entity recognition (NER)*



La Bombilla

Rúa Torreiro, 6, A Coruña

Escribir una reseña

4,3 5.767 reseñas

Las reseñas no se verifican. ⓘ

Los usuarios suelen mencionar

Todas

emblemático 39

plástico 34

mítico 30

tradición 28

esencia 14

caldo gallego 11

estrella galicia 11

milanesa 10

servilletero 9

clásico 8

Ordenar por

Más relevantes

Más recientes

Más alta

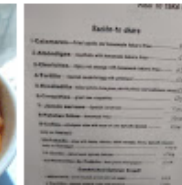
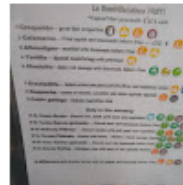
Más baja



5.753 fotos

Hace 8 meses

Fue una visita rápida de cañas y tapas. Una lástima que las tapas de milanesa y calamares estaban frías. Rico sabor pero mal la temperatura. A los callos les faltaba sabor y tenían muchas partes de cerdo que eran solo grasa. La ensaladilla estaba muy rica.



2

Extraída de Google Reviews

Aplicaciones de *named-entity recognition (NER)*

HISTORY: Patient is a 21-year-old white woman who presented with a chief complaint of chest pain SYMPTOM X .

She had been previously diagnosed with hyperthyroidism DISEASE X .

Upon admission, she had complaints of constant left sided chest pain SYMPTOM X that radiated to her left arm.

She had been experiencing palpitations SYMPTOM X and tachycardia SYMPTOM X .

She had no diaphoresis SYMPTOM X , no nausea SYMPTOM X , vomiting SYMPTOM X , or dyspnea SYMPTOM X .

She had a significant TSH of 0.004 and a free T4 of 19.3.

Normal ranges for TSH and free T4 are 0.5-4.7 μ IU/mL and 0.8-1.8 ng/dL, respectively.

Her symptoms started four months into her pregnancy as tremors SYMPTOM X , hot flashes SYMPTOM X , agitation SYMPTOM X , and emotional inconsistency SYMPTOM X .

She gained 16 pounds during her pregnancy and has lost 80 pounds afterwards.

She complained of sweating SYMPTOM X , but has experienced no diarrhea SYMPTOM X and no change in appetite.

She was given isosorbide mononitrate CHEMICAL X and IV steroids in the ER.

FAMILY HISTORY: Diabetes, Hypertension, Father had a Coronary Artery Bypass Graph (CABG) at age 34.

MEDICATIONS: Citalopram CHEMICAL X 10mg DOSAGE X once daily for depression DISEASE X ; low dose tramadol PRN pain.

PHYSICAL EXAMINATION: Temperature 98.4; Pulse 123; Respiratory Rate 16; Blood Pressure 143/74. HEENT: She has exophthalmos and could not close her lids completely. Cardiovascular: tachycardia. Neurologic: She had mild hyperreflexiveness.

LAB: All labs within normal limits with the exception of Sodium 133, Creatinine 0.2, TSH 0.004, Free T4 19.3 EKG showed sinus tachycardia with a rate of 122.

Urine pregnancy test was negative.

HOSPITAL COURSE: After admission, she was given propranolol CHEMICAL X at 40mg DOSAGE X daily and continued on telemetry.

On the 2nd day of treatment, the patient still complained of chest pain SYMPTOM X .

EKG again showed tachycardia SYMPTOM X .

Propranolol CHEMICAL X was increased from 40mg DOSAGE X daily to 60mg DOSAGE X twice daily.

A I-123 thyroid uptake scan demonstrated an increased thyroid uptake of 90% at 4 hours and 94% at 24 hours.

Extraída de: <https://www.persistent.com/blogs/building-named-entity-recognition-models-for-healthcare/>

Modelos neuronales de etiquetado de secuencia

Long Short-Term Memory networks (LSTMs): Modelos con potencial capacidad para mantener contexto ilimitado, basados en redes neuronales recurrentes y capaces de mantener memoria a largo plazo.

Embeddings aleatorias vs. preentrenadas: Compararemos modelos con embeddings inicializadas aleatoriamente y modelos con embeddings preentrenadas del Word2Vec de Google.

El objetivo es aprender a implementar modelos que tratan el contexto de diferentes maneras. La capacidad para manejar el contexto correctamente es uno de los aspectos más importantes para poder obtener buenos rendimientos en muchas tareas de PLN.

Modelos neuronales de etiquetado de secuencia

Metodología estándar para la creación de los modelos.

Fase entrenamiento/desarrollo:

- Un conjunto de entrenamiento (train.txt) y otro de desarrollo (dev.txt), comúnmente en texto plano, con anotaciones a nivel de palabra.
- Un modelo supervisado capaz de aprender a partir de dichos datos.

Evaluación:

- Un conjunto de test (test.txt) que el modelo no haya visto antes, para estimar su rendimiento en entornos reales.

Long short-term memory networks - Entrenamiento

Mary has a cat

PROPN VERB DET NOUN



17 56 28 100

1 3 6 2

Spiderman exists

PROPN VERB



7 324

1 3

I am real

PRONOUN VERB ADJ



98 76 3

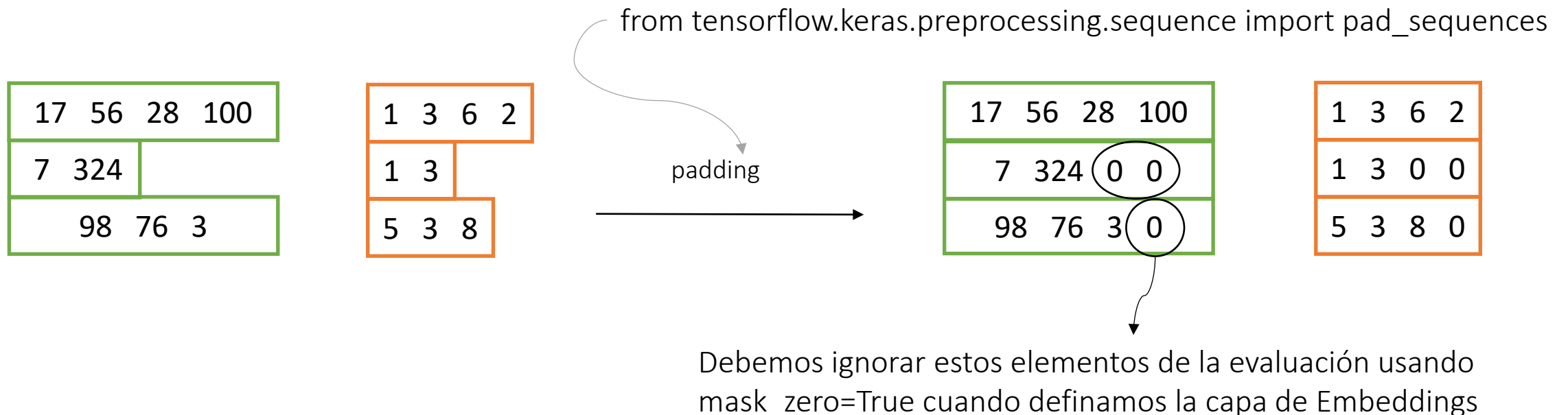
5 3 8

Long short-term memory networks - Entrenamiento

En un batch, tendremos oraciones con diferentes longitudes.

Todas las oraciones en un batch deben tener la misma longitud para que puedan ser enviadas a un modelo de keras

La solución es similar al caso del modelo con feed-forward networks: incluir padding.



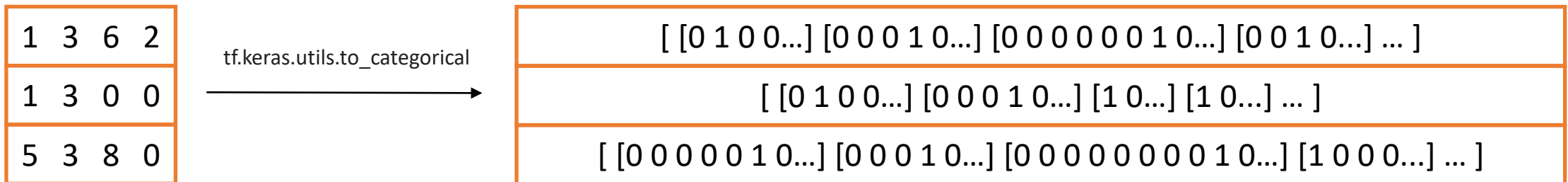
Long short-term memory networks - Entrenamiento

Es un problema de clasificación multi-clase.

Optimizaremos el modelo con categorical_crossentropy loss:

https://www.tensorflow.org/api_docs/python/tf/keras/metrics/categorical_crossentropy

Para ello, necesitaremos transformar nuestra etiqueta (inicialmente representada como una cadena) en un one-hot vector que identifica la clase que se debe predecir.



Alternativamente a categorical_crossentropy loss también podemos usar una sparse categorical crossentropy loss (en ese caso no necesitaremos la función to_categorical):

https://www.tensorflow.org/api_docs/python/tf/keras/losses/SparseCategoricalCrossentropy

Long short-term memory networks - Entrenamiento

Capa de entrada: https://www.tensorflow.org/api_docs/python/tf/keras/Input

Capa de embeddings: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Embedding

Capa LSTM: https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM

Capa bidireccional: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Bidirectional

Capa Dense: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Dense

Capa TimeDistributed [importante!]:
https://www.tensorflow.org/api_docs/python/tf/keras/layers/TimeDistributed

tf.keras.layers.TimeDistributed

Un wrapper para aplicar una capa a cada timestep de una entrada temporal (como por ejemplo una oración).

La dimension en el eje 1 es la que se considera como la dimension temporal.

En nuestro caso, la forma de la entrada para nuestro etiquetador es (batch_size, max_sentence_length, word_embedding_size).

Dada una capa TimeDistributed, aplicará la capa 'envuelta' a todos los elementos de entrada de la secuencia.

tf.keras.layers.TimeDistributed

model = ...

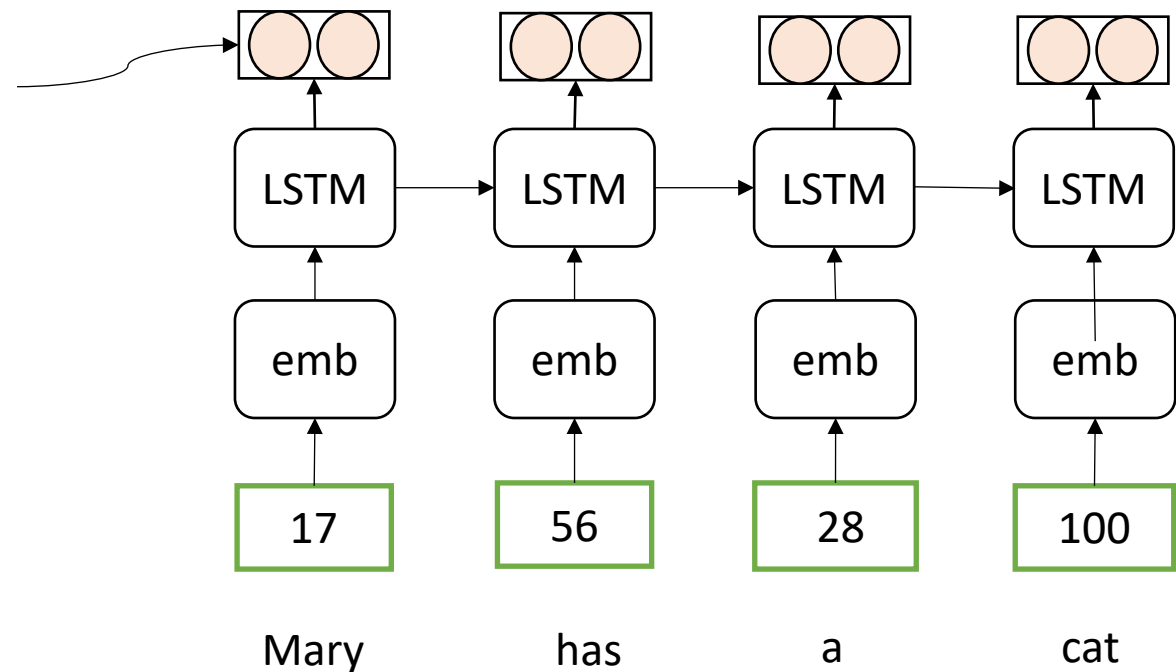
...

Añadir la capa de Embeddings

Añadir la capa de LSTMs

...

La salida de la LSTM debe ser un vector para cada palabra (chequear el parámetro `return_sequences` de la capa LSTM).



tf.keras.layers.TimeDistributed

model = ...

...

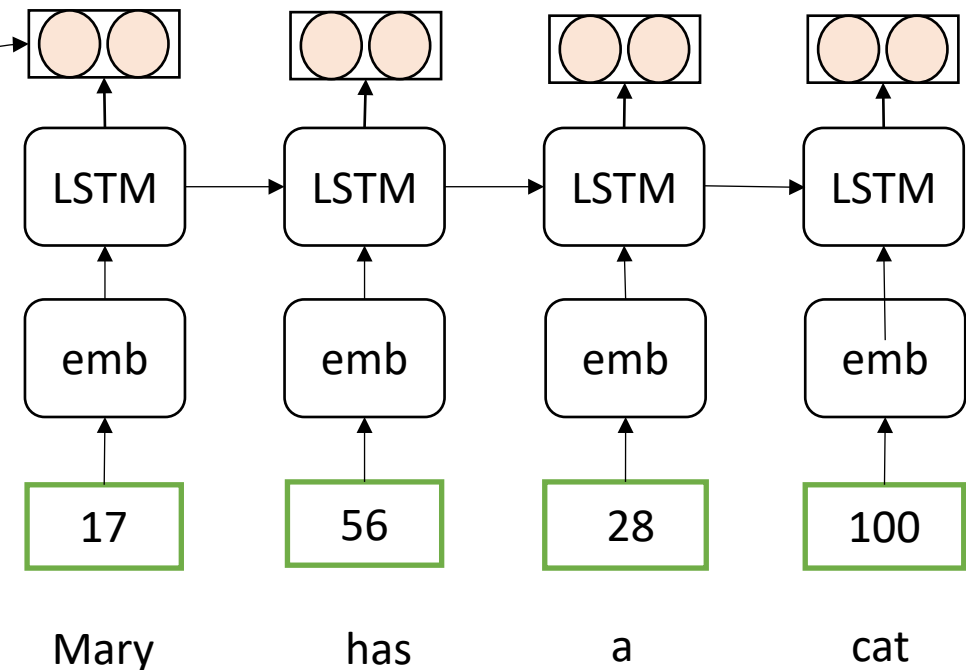
Añadir la capa de Embeddings

Añadir la capa de LSTMs

...

La salida de la LSTM debe ser un vector para cada palabra (chequear el parámetro `return_sequences` de la capa LSTM).

Sin embargo, la capa Dense que viene a continuación no está pensada para ser aplicada sobre secuencias, ¿cómo podemos obtener entonces fácilmente la etiqueta para todos los elementos de entrada 'de una vez'? Usando el wrapper `TimeDistributed`, que hace justamente eso.



tf.keras.layers.TimeDistributed

model = ...

...

We add the embedding layer

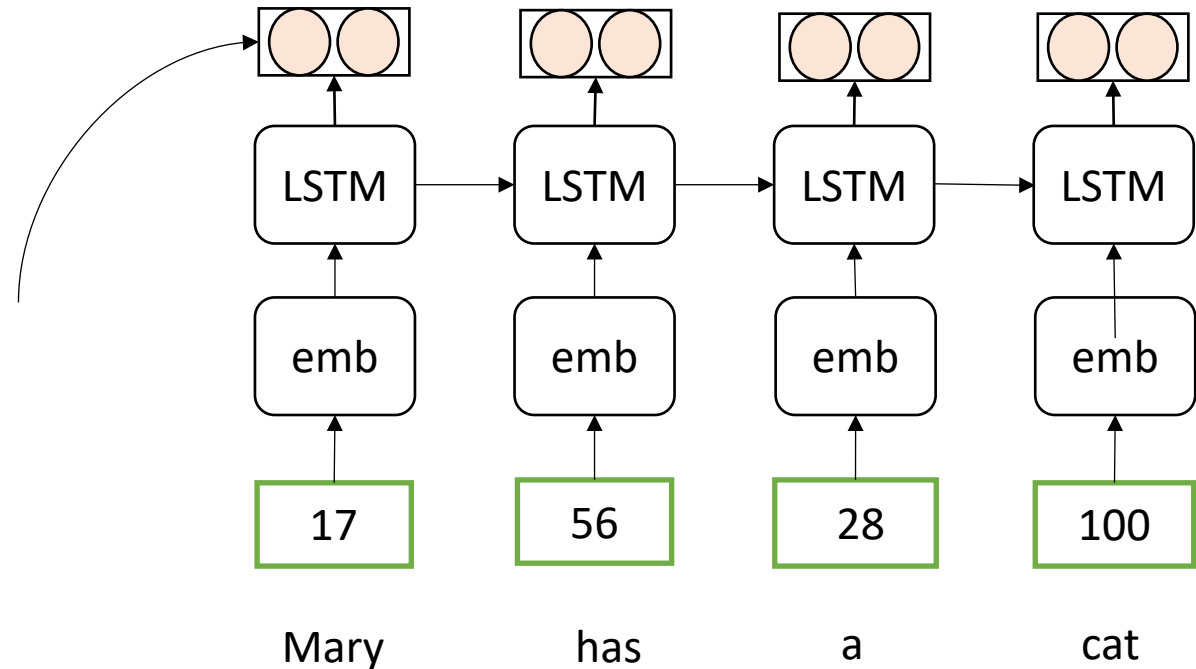
We add the LSTM layer

...

```
model.add(TimeDistributed(Dense(nlabels,  
activation='softmax')))
```

...

```
model.fit(...)
```



tf.keras.layers.TimeDistributed

model = ...

...

We add the embedding layer

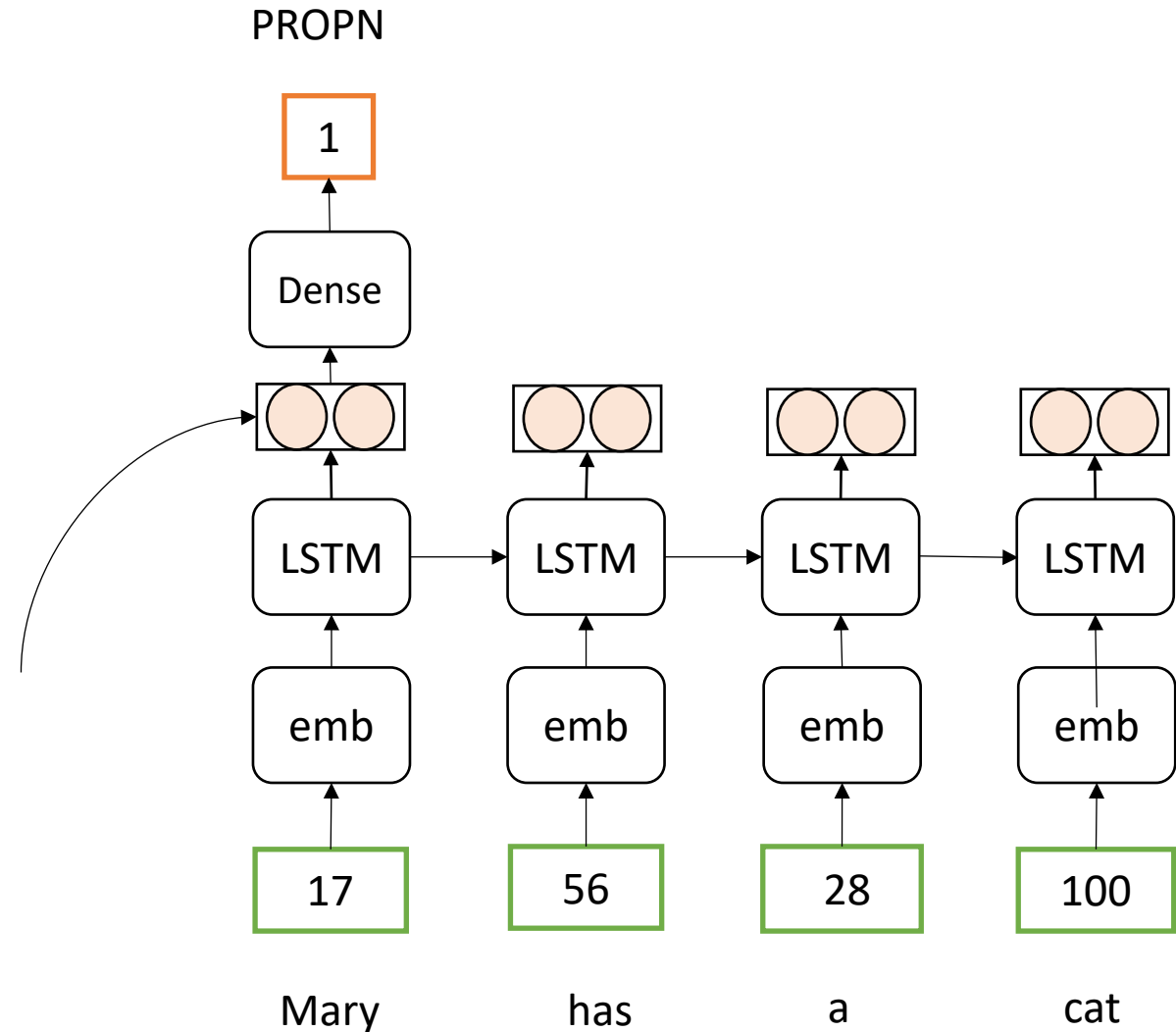
We add the LSTM layer

...

```
model.add(TimeDistributed(Dense(nlabels,  
activation='softmax')))
```

...

```
model.fit(...)
```



tf.keras.layers.TimeDistributed

model = ...

...

We add the embedding layer

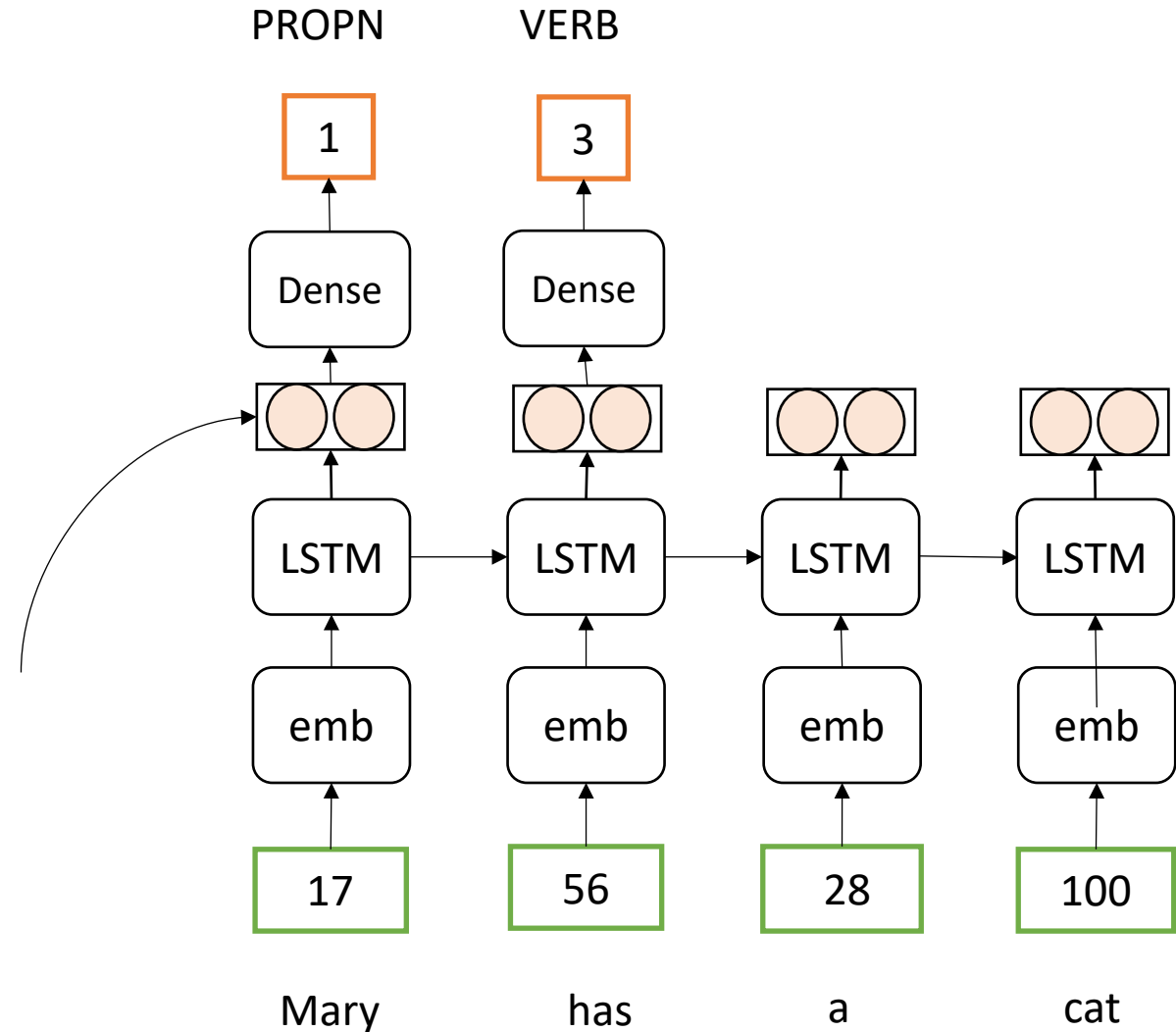
We add the LSTM layer

...

```
model.add(TimeDistributed(Dense(nlabels,  
activation='softmax')))
```

...

```
model.fit(...)
```



tf.keras.layers.TimeDistributed

model = ...

...

We add the embedding layer

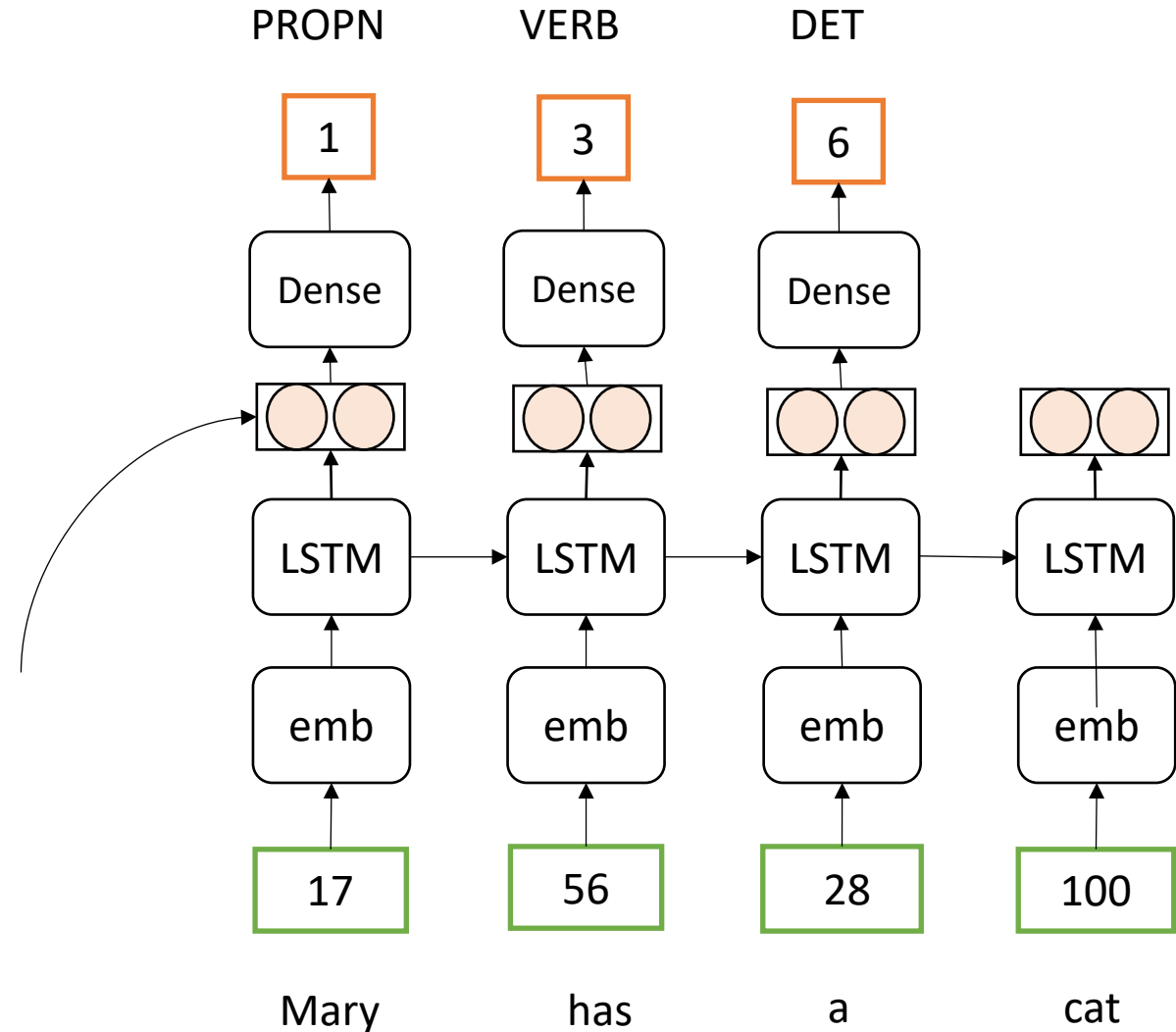
We add the LSTM layer

...

```
model.add(TimeDistributed(Dense(nlabels,  
activation='softmax')))
```

...

```
model.fit(...)
```



tf.keras.layers.TimeDistributed

model = ...

...

We add the embedding layer

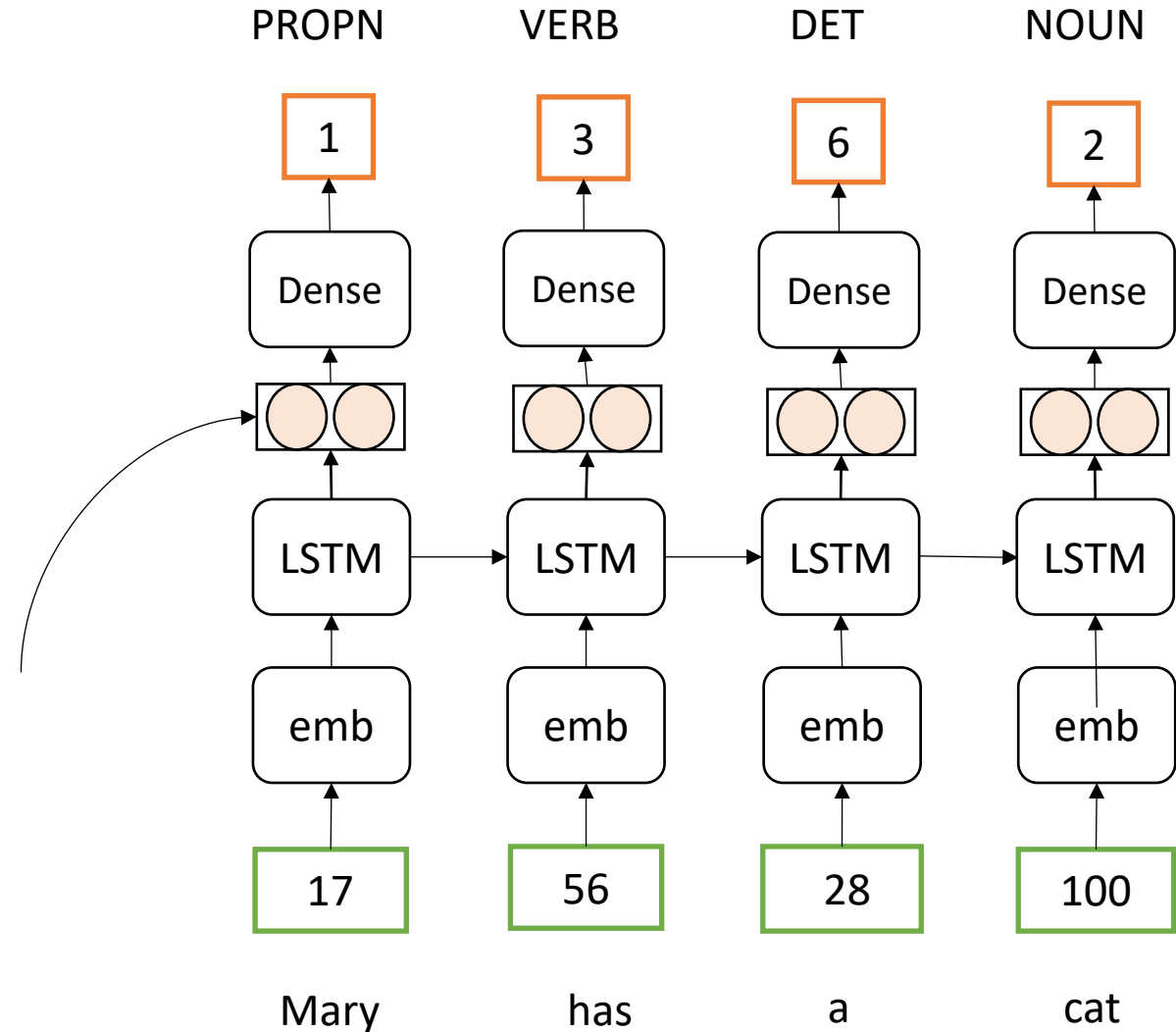
We add the LSTM layer

...

```
model.add(TimeDistributed(Dense(nlabels,  
activation='softmax')))
```

...

```
model.fit(...)
```



Evaluación

PoS tagging: Usaremos la accuracy reportada por el propio método `evaluate()` de los modelos de keras.

Named-entity recognition: Además de la accuracy, como en el caso de PoS tagging, deberán obtenerse la F1-score para cada tipo de entidad para las categorías de “ent_type” “partial” “exact” y “strict”. Para ello debe usarse la librería de Python `nervaluate`, que debéis instalar (por ejemplo a través de pip): [nervaluate · PyPI](#)

Datasets que emplearemos

Análisis morfológico: Una version del corpus [ud-en-partut](#) ya procesada para PoS tagging y dividida en conjuntos de entrenamiento, desarrollo y test.

Reconocimiento de entidades nombradas: Una version del dataset [MITRestaurant](#), ya dividido en conjuntos de entrenamiento, desarrollo y test. Usad el material incluido con esta práctica, no el dataset original, que solo contiene conjuntos de entrenamiento y test.

Algunas recomendaciones prácticas.

Transformar las palabras en una lista de ids. Como para los datasets propuestos las palabras han sido ya previamente tokenizadas, solo necesitamos implementar una función ad-hoc que asocie palabras a IDs.

Rellenar las muestras de entrada con padding, de manera que todas las muestras de entrenamiento de un batch tengan la misma longitud. Esto es requerido por keras para poder trabajar con batches.

Convertir también las etiquetas de salida a IDs numéricos, para que el modelo pueda ser entrenado con la función fit.

Algunas recomendaciones prácticas.

En el caso de NER, la etiqueta O es claramente mayoritaria. Ello puede causar algunas dificultades para que el modelo aprenda correctamente a identificar las entidades. Por ello, se recomienda asignar un peso a cada posible etiqueta. En el caso de tareas de etiquetado de secuencias, esto puede hacerse con el parámetro `sample_weights` a través de la función `fit`, o si se usa `tf.data.Dataset.from_tensor_slices` para preparar el conjunto de datos, se puede añadir el paso para cada muestra como un elemento más de la tupla.

Algunas recomendaciones prácticas.

Se puede entrenar los modelos con GPU para hacerlo de manera más rápido.

Para ello, es posible usar los recursos que libremente da <https://colab.research.google.com/>

En materiales/gpu.ipynb se explica como seleccionar que los modelos se ejecuten en GPU y comprobar que el código Python está detectando la GPU.

Algunas recomendaciones prácticas.

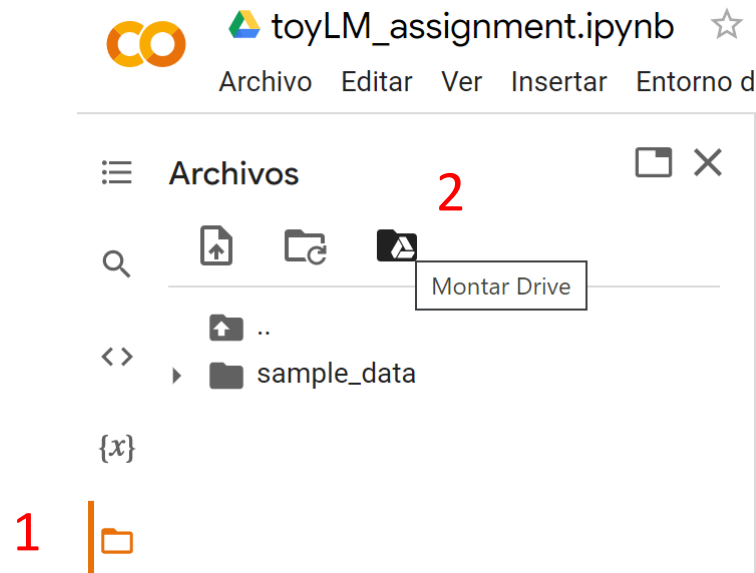
Se puede entrenar los modelos con GPU para hacerlo de manera más rápido.

Para ello, es posible usar los recursos que libremente da <https://colab.research.google.com/>

En materiales/gpu.ipynb se explica como seleccionar que los modelos se ejecuten en GPU y comprobar que el código Python está detectando la GPU.

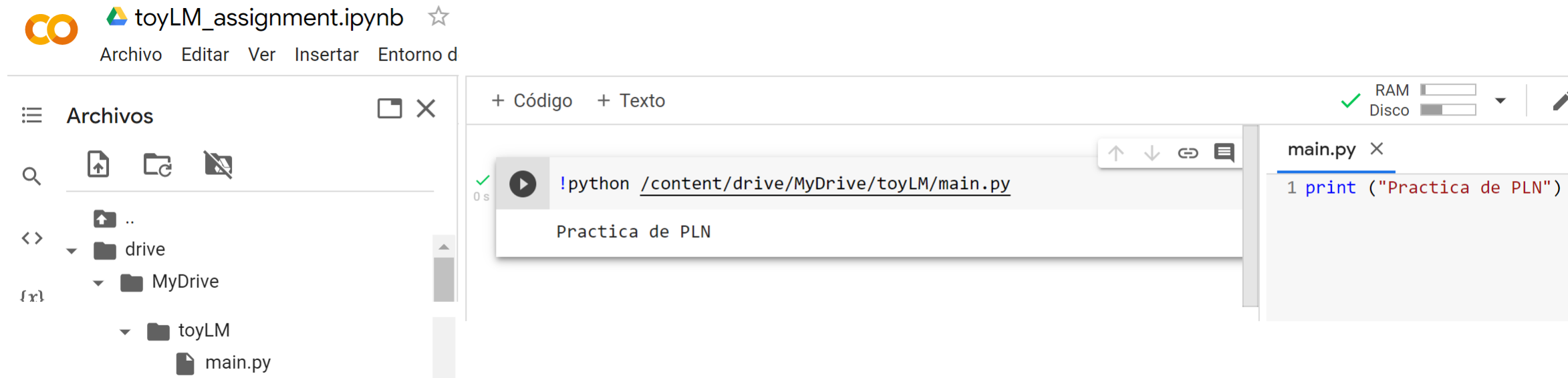
Cargar y editar en Colab un Proyecto externo Python

1. Crear en la cuenta de Google Drive un directorio para el proyecto.
Por ejemplo en MyDrive/PLE.
2. Crear un notebook (ple_assignment) y enlazarlo con la Google Drive:



Cargar y editar un proyecto externo en Python en Colab

3. Probar a ejecutar el fichero main.py en MyDrive/assignment tal y como se indica en la siguiente figura.



4. Comenzar a implementar la práctica. Se recomienda implementar y depurar en local y usar Colab principalmente para entrenar los modelos.