



SPAIN AI  
AI WORKSHOPS

WORKSHOP GRATUITO EN ESPAÑOL

# NLP de 0 a 100 con Hugging Face

Apúntate en **eventbrite**



 7 SESIONES MARTES (quincenal)  
DEL 13/07 AL 5/10 (18-18:40 CET)

Imparten:

- María Grandury
- Manuel Romero
- Omar Sanseviero
- Lewis Tunstall

¡SÍGUENOS!

@NLP\_en\_ES

@Spain\_AI\_

# Enlaces útiles

---



[@nlp-en-es/nlp-de-cero-a-cien](#)



[#nlp-de-cero-a-cien](#)



[playlist: NLP de 0 a 100 con 🙌](#)



[@spain\\_ai](#)

[@nlp en es](#)



[@company/spainai](#)

[@company/nlp-en-es](#)

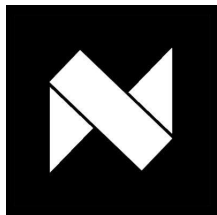


[spain-ai.com](#)

[nlp-en-es.org](#)




# Sobre Manuel Romero [@mrm8488](#)




[NLP/NLG Engineer at Narrativa](#)

@NLP\_en\_ES

 **Hugging Face**


**Models** **Datasets** **Resources** **Solutions** **Pricing** **Log In** **Sign Up**



**Manuel Romero** PRO  
[mrm8488](#)

**Homepage**  
<https://mrm8488.github.io>

**Research interests**  
None yet

**Organizations**  


**Spaces**  
None yet

**Models** 231

<b>mrm8488/AfricanBERTa</b> Fill-Mask · Updated May 20 · 8	<b>mrm8488/CodeBERTaPy</b> Fill-Mask · Updated May 20 · 60
<b>mrm8488/CodeGPT-small-finetuned-python-token-complet...</b> Text Generation · Updated May 23 · 130	<b>mrm8488/GPT-2-finetuned-CORD19</b> Text Generation · Updated May 23 · 148
<b>mrm8488/GPT-2-finetuned-CRD3</b> Text Generation · Updated May 23 · 88	<b>mrm8488/GPT-2-finetuned-common_gen</b> Text Generation · Updated May 23 · 304
<b>mrm8488/GPT-2-finetuned-covid-bio-medixiv</b> Text Generation · Updated Aug 25 · 306	<b>mrm8488/GuaPeTe-2-tiny-finetuned-TED</b> Text Generation · Updated May 23 · 43
<b>mrm8488/GuaPeTe-2-tiny-finetuned-eubookshop</b> Text Generation · Updated May 23 · 21	<b>mrm8488/GuaPeTe-2-tiny-finetuned-spa-constitution</b> Text Generation · Updated May 23 · 21
<b>mrm8488/GuaPeTe-2-tiny</b> Text Generation · Updated May 23 · 36	<b>mrm8488/HindiBERTa</b> Fill-Mask · Updated May 20 · 121
<b>mrm8488/ManuERT-foi-xqua</b> Question Answering · Updated May 20 · 31	<b>mrm8488/RobERTinha</b> Fill-Mask · Updated May 20 · 21
<b>mrm8488/RoBasquERTA</b> Fill-Mask · Updated May 20 · 29	<b>mrm8488/RuPERTa-base-finetuned-nez</b> Token Classification · Updated May 20 · 347 · 1



# Plan

---

- Pre-training (teoría)
- Cómo hacer Pre-training con HuggingFace
- [Fine-tuning un GPT-2 en Español para generar texto](#)



# Pre-training: ¿Por qué?

---

→ Adquisición de conocimiento “general” de cómo funciona el lenguaje.

- ◆ Corpus/dataset “enorme”

- ◆ Gran consumo de recursos HW

- ◆ BERT:

  - Wikipedia + BookCorpus (16 GB)

  - 4 días con de 4 a 16 Cloud TPUs

→ Adaptabilidad a tareas específicas (fine-tuning)

- ◆ Eficacia

- ◆ Rapidez



# Pre-training: ¿Cómo?

---

→ 2 estrategias:

◆ MLM:

- Se “enmascara” un porcentaje de “tokens” (15%) de la frase de entrada y el modelo intenta descifrar esos “tokens”.

◆ NSP:

- Se generan frases pares de frases de las cuáles unas pertenecen al mismo contexto y otras no (NLI, QA).



# Pre-training: ¿Cómo? II

---

→ 2 estrategias:

◆ MLM:

- Input: El hombre fue a la **[MASK1]** . Compró un **[MASK2]** de leche.
- Labels: [MASK1] = tienda; [MASK2] = litro

◆ NSP:

- Frase A: El hombre fue a la tienda.
- Frase B: Compró un litro de leche.
  - Label: **IsNextSentence**
- Frase A: El hombre fue a la tienda.
- Frase B: En verano suele hacer calor.
  - Label: **NotNextSentence**



# Pre-training: Benchmarks

Glue Benchmark Leaderboard

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
BERT <sub>BASE</sub>	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
BERT <sub>LARGE</sub>	<b>86.7/85.9</b>	<b>72.1</b>	<b>91.1</b>	<b>94.9</b>	<b>60.5</b>	<b>86.5</b>	<b>89.3</b>	<b>70.1</b>	<b>81.9</b>





# Pre-training: Conclusiones (BERT)

---

- Las mejoras que se obtienen al aplicar “**transfer learning**” para una tarea determinada demuestran que el pre-training (no supervisado y sobre un gran corpus) es una parte clave para los sistemas de NLU.
- Beneficia especialmente a tareas donde los recursos son limitados (Dataset/HW).



# Pre-training: Otras estrategias

---

→ Estrategias:

◆ RoBERTa: (BERT optimizado)

- Enmascaramiento dinámico de tokens durante el entrenamiento
- No hay NSP
- Lotes (“batches”) más grandes y mayor corpus (BERT x 10)

◆ ELECTRA:

- Enfoque “GAN” (generador y discriminador)
- Generador: reemplaza tokens de la entrada (entrenado como MLM)
- Discriminador: intenta detectar qué tokens son “fake” (el modelo en sí)
- Ventaja: Resultados SOTA de manera eficiente a nivel de recursos HW



# Pre-training: Pasos

---

- Elegir una arquitectura
  - ◆ BERT, RoBERTa, ELECTRA, GPT-2, Seq2Seq
- Elegir un corpus
  - ◆ Tamaño y calidad (sampling?)
- Crear un vocabulario y tokenizar el corpus
  - ◆ Tipo de tokenizer (BPE, sentencepiece)
- Entrenar el modelo
  - ◆ Establecer hiperparámetros
    - Tamaño del “batch”, “learning rate”, tiempo, etc



# Pre-training: Pasos con HuggingFace

---

- Elegir una arquitectura
  - ◆ HF implementa las arquitecturas más conocidas (*config.json*)
- Elegir un corpus
  - ◆ HF [Datasets](#) (streaming mode)
- Crear un vocabulario y tokenizar el corpus
  - ◆ HF [Tokenizers](#) (implementación en Rust)
- Entrenar el modelo
  - ◆ HF [examples/Notebooks](#)
  - ◆ [Colab de ejemplo](#)



# Pre-training: Pasos con [HuggingFace](#) II

```
from datasets import load_dataset
from tokenizers import trainers, Tokenizer, normalizers, ByteLevelBPETokenizer

# load dataset
dataset = load_dataset("oscar", "unshuffled_deduplicated_no", split="train")

# Instantiate tokenizer
tokenizer = ByteLevelBPETokenizer()

def batch_iterator(batch_size=1000):
    for i in range(0, len(dataset), batch_size):
        yield dataset[i: i + batch_size]["text"]

# Customized training
tokenizer.train_from_iterator(batch_iterator(), vocab_size=50265, min_frequency=2, special_tokens=[
    "<s>",
    "<pad>",
    "</s>",
    "<unk>",
    "<mask>",
])

# Save files to disk
tokenizer.save("./tokenizer.json")
```



## Pre-training: Pasos con [HuggingFace](#) III

---

```
from transformers import RobertaConfig

config = RobertaConfig.from_pretrained("roberta-base", vocab_size=50265)
config.save_pretrained("./")
```



# Pre-training: Pasos con [HuggingFace](#) IV

```
./run_mlm_flax.py \
  --output_dir="./" \
  --model_type="roberta" \
  --config_name="./" \
  --tokenizer_name="./" \
  --dataset_name="oscar" \
  --dataset_config_name="unshuffled_deduplicated_no" \
  --max_seq_length="128" \
  --weight_decay="0.01" \
  --per_device_train_batch_size="128" \
  --per_device_eval_batch_size="128" \
  --learning_rate="3e-4" \
  --warmup_steps="1000" \
  --overwrite_output_dir \
  --num_train_epochs="18" \
  --adam_beta1="0.9" \
  --adam_beta2="0.98" \
  --logging_steps="500" \
  --save_steps="2500" \
  --eval_steps="2500" \
  --push_to_hub
```



# Pre-training: BERTIN (caso práctico)

---

- Arquitectura: RoBERTa
- Corpus: *mC4 es* (sampling)
- Entrenamiento: [Flax/Jax Google & HuggingFace Community Event](#) (TPUs VMs)

