TEMA 5 – HOJA DE EJERCICIOS II

Se plantean ejercicios para seguir practicando con la arquitectura basada en Transformers, en este caso, utilizando modelos de lenguaje.

Ejercicio 1

Se propone un ejercicio para aprender a ajustar (fine-tuning) el modelo pre-entrenado BERT para la tarea clasificación de textos. En particular, para clasificar comentarios de películas como positivas o negativas. Para ello, se puede utilizar un subconjunto del dataset "rotten_tomatoes".

Los pasos que hay que seguir son:

1) Instalar la librería transformer de Hugging Face.
2) Cargar los datos y sacar las labels.

   ```
   Labels:  ['neg', 'pos'] , número de labels:  2
   ```

3) Tokenizar. BERT admite 512 tokens como tamaño de entrada. Por tanto, hay que comprobar cuál es el tamaño de los textos de entrada para tener en cuenta si hay que hacer padding y/o truncation. Hay que establecer la longitud máxima.

   Decidir el modelo que se va a utilizar de Bert (multilingüe, uncased, etc.). En este ejercicio usaremos el modelo "bert-base-uncased".

   ```python
   from transformers import AutoTokenizer

   model_id = "bert-base-uncased"

   tokenizer = AutoTokenizer.from_pretrained(model_id)
   ```

4) Fine-tuning del modelo. La clase AutoModelForSequenceClassification permite cargar el modelo para extenderlo para la tarea de clasificación de secuencias. En concreto, este modelo extendido tiene una última capa softmax que obtiene una probabilidad por cada clase. En nuestro caso, es un problema de clasificación binaria.

Además, del modelo es necesario, indicar el número de clases (labels).

```
from transformers import AutoModelForSequenceClassification
model = AutoModelForSequenceClassification.from_pretrained("bert-base-uncased", num_labels=NUM_LABELS)
```

5) Hiperparámetros

Hay que definir los hiperparámetros que se emplearán durante el proceso de entrenamiento. Algunos de estos hiperpárametros son el número de epochs para el entrenamiento, el tamaño del lote (batch), la ratio de aprendizaje (learning rate), etc. Puedes experimentar con estos hiperparámetros para encontrar la configuración óptima para tu tarea.

Se puede comenzar trabajando con los hiperparámetros por defecto.

Para ello vamos a crear un objeto de la clase TrainingArguments, que incluye todos los hiperparámetros (ya inicializados con valores por defecto) que se pueden ajustar.

A continuación, se muestra un código que permite ver la configuración:

```
from transformers import TrainingArguments
args = TrainingArguments(output_dir="./outputs")
args
```

6) Decidir las métricas para evaluar

7) Entrenar el modelo

Se configurar los hiperparámetros que se pasaran como argumento a un objeto de tipo Trainer (clase que nos ayuda a configurar el entrenamiento).

```
from transformers import Trainer

trainer = Trainer(
    model = model,            # modelo que será ajustado
    #train_dataset = encoded_data['train'], # conjunto training
    train_dataset = small_train_dataset,
    #eval_dataset = encoded_data['validation'],   # conjunto de validación
    eval_dataset = small_train_dataset,

    args = args,     # hiperparámetros
    compute_metrics=compute_metrics,    # función para computar las métricas
)
```

8) Realizar la evaluación

El modelo entrenado se utilizar para predecir las clases de los textos del conjunto de test. Sobre la salida del modelo se aplica la función softmax, que calcula la probabilidad de cada clase y para devolver la que tiene mayor probabilidad utilizamos la función argmax.

```
def get_prediction(text):
    # prepara el texto, aplicamos la misma tokenización que la utilizada en el training
    inputs = tokenizer(text, padding="max_length", max_length=MAX_LENGTH, truncation= True, return_tensors="pt").to("cuda")

    # aplicamos el modelo
    pred = model(**inputs).logits

    # obtenemos la probabilidad para cada clase
    probs = pred.softmax(1)
    # devolvemos la mayor
    return probs.argmax().item()
```

Se quiere saber la predicción para los textos del conjunto test (estos textos que no han sido utilizados durante el entrenamiento).

Nota: para hacer este ejercicio se proporciona un notebook casi al completo. Únicamente falta hacer el punto 8), la inferencia sobre el conjunto de test.

Se puede probar con diferentes configuraciones de hiperparámetros.

Ejercicio 2

Realizar lo mismo que en el ejercicio 1, pero se quiere hacer inferencia sobre textos nuevos, ¿cuál sería la salida para las siguientes frases?

'i hate you too much'

'I did not like the film at all'

'I loved the movie'

En este caso hay que tokenizer cada frase del mismo modo que se ha tokenizado el resto del corpus y, además, hay que indicar truncation=True como parámetro.

¿Está acertando en los resultados?

Ejercicio 3

Se pide hacer lo mismo que en los ejercicios anteriores, pero en este caso se pide ajustar otros modelos diferentes, por ejemplo:

- bert-base-cased
- distilbert-base-uncased
- albert-base-v2
- xlm-roberta-base

Compara los resultados obtenidos con los diferentes modelos. ¿Cuál de ellos es el que obtiene mejor valor de F1? ¿Cuál es el más rápido?