Multiclass Text Classification with

Feed-forward Neural Networks and Word Embeddings

María Fernanda Pérez Ruiz A01742101

First, we will do some initialization.

En este primer chunk de código preparamos el entorno para el pipeline de procesamiento y entrenamiento en NLP, primeramente importamos la librerias claves necesarias: random, torch, numpy (np) y pandas, van a ser necesarias para poder trabajar y manipular los datos y poder entrenar el modelo en PyTorch, se incluye tambien tqdm que nos va a ayudar a facilitar el seguimiento visual del progreso en las operaciones de procesamiento con pandas.

Despues en el código se verifica si es que hay un GPU disponible paara el entrenamiento, si es que encuentra una le asigna el dispositivo cuda. Despues establecemos nuestra random seed en 1234 para que al volver correr el ejercicio obtengamos los mismos resultados.

Finalmente se imprimen el dispositivo seleccionado y la semilla que ya preestablecimos para confirmar la configuración del entorno de entrenamiento.

```
import random
import torch
import numpy as np
import pandas as pd
from tqdm.notebook import tqdm

# enable tqdm in pandas
tqdm.pandas()

# set to True to use the gpu (if there is one available)
use_gpu = True

# select device
device = torch.device('cuda' if use_gpu and torch.cuda.is_available()
else 'cpu')
print(f'device: {device.type}')

# random seed
seed = 1234

# set random seed
if seed is not None:
    print(f'random seed: {seed}')
```

```
    random.seed(seed)
    np.random.seed(seed)
    torch.manual_seed(seed)

device: cuda
random seed: 1234
```

We will be using the AG's News Topic Classification Dataset. It is stored in two CSV files: `train.csv` and `test.csv`, as well as a `classes.txt` that stores the labels of the classes to predict.

First, we will load the training dataset using pandas and take a quick look at how the data.

En este chunk de código cargamos con pandas el conjunto de datos de entrenamiento para la clasificación. El primer archivo que lee es el de train al cual le asignamos el nombre a las columnas: 'class index', 'title', 'description'.

```
train_df =
pd.read_csv('/kaggle/input/ag-news-classification-dataset/train.csv',
header=None)
train_df.columns = ['class index', 'title', 'description']
train_df
```

```
        class index                                              title
\
0       Class Index                                              Title

1                 3  Wall St. Bears Claw Back Into the Black (Reuters)

2                 3  Carlyle Looks Toward Commercial Aerospace (Reu...

3                 3    Oil and Economy Cloud Stocks' Outlook (Reuters)

4                 3  Iraq Halts Oil Exports from Main Southern Pipe...

...             ...                                                ...

119996            1  Pakistan's Musharraf Says Won't Quit as Army C...

119997            2                  Renteria signing a top-shelf deal

119998            2                    Saban not going to Dolphins yet

119999            2                                Today's NFL games

120000            2                        Nets get Carter from Raptors


                                           description
0                                          Description
1       Reuters - Short-sellers, Wall Street's dwindli...
```

```
2       Reuters - Private investment firm Carlyle Grou...
3       Reuters - Soaring crude prices plus worries\ab...
4       Reuters - Authorities have halted oil export\f...
...                                                   ...
119996   KARACHI (Reuters) - Pakistani President Perve...
119997  Red Sox general manager Theo Epstein acknowled...
119998  The Miami Dolphins will put their courtship of...
119999  PITTSBURGH at NY GIANTS Time: 1:30 p.m. Line: ...
120000  INDIANAPOLIS -- All-Star Vince Carter was trad...

[120001 rows x 3 columns]
```

The dataset consists of 120,000 examples, each consisting of a class index, a title, and a description. The class labels are distributed in a separated file. We will add the labels to the dataset so that we can interpret the data more easily. Note that the label indexes are one-based, so we need to subtract one to retrieve them from the list.

En este chunk de código se cargan las etiquetas de las clases tomadas del archivo classes.txt, el cual son los nombres de las categorías de noticias correspondientes a cada índice de clase , despues el dataframe train_df se reindexa para reiniciar los índices después de haber tomado una muestra en el paso anterior. La columna 'class index' la vamos a hacer entero para asegurarnos del formato. Con la función de map vamos a asociar cada índice de clase con su respectiva etiqueta textual desde labels, se ocupan ajustar los índices porque los valores en el archivos empiezan desde 1. Por ultimo insertamos la nueva columna llamada 'class' que incluye los nombres de las clases.

```
labels =
open('/kaggle/input/classes-txt/classes.txt').read().splitlines()
train_df = train_df.drop(0).reset_index(drop=True)
train_df['class index'] = train_df['class index'].astype(int)
classes = train_df['class index'].map(lambda i: labels[i-1])
train_df.insert(1, 'class', classes)
train_df
```

```
        class index     class  \
0                 3  Business
1                 3  Business
2                 3  Business
3                 3  Business
4                 3  Business
...             ...       ...
119995            1     World
119996            2    Sports
119997            2    Sports
119998            2    Sports
119999            2    Sports

                                                    title  \
0       Wall St. Bears Claw Back Into the Black (Reuters)
```

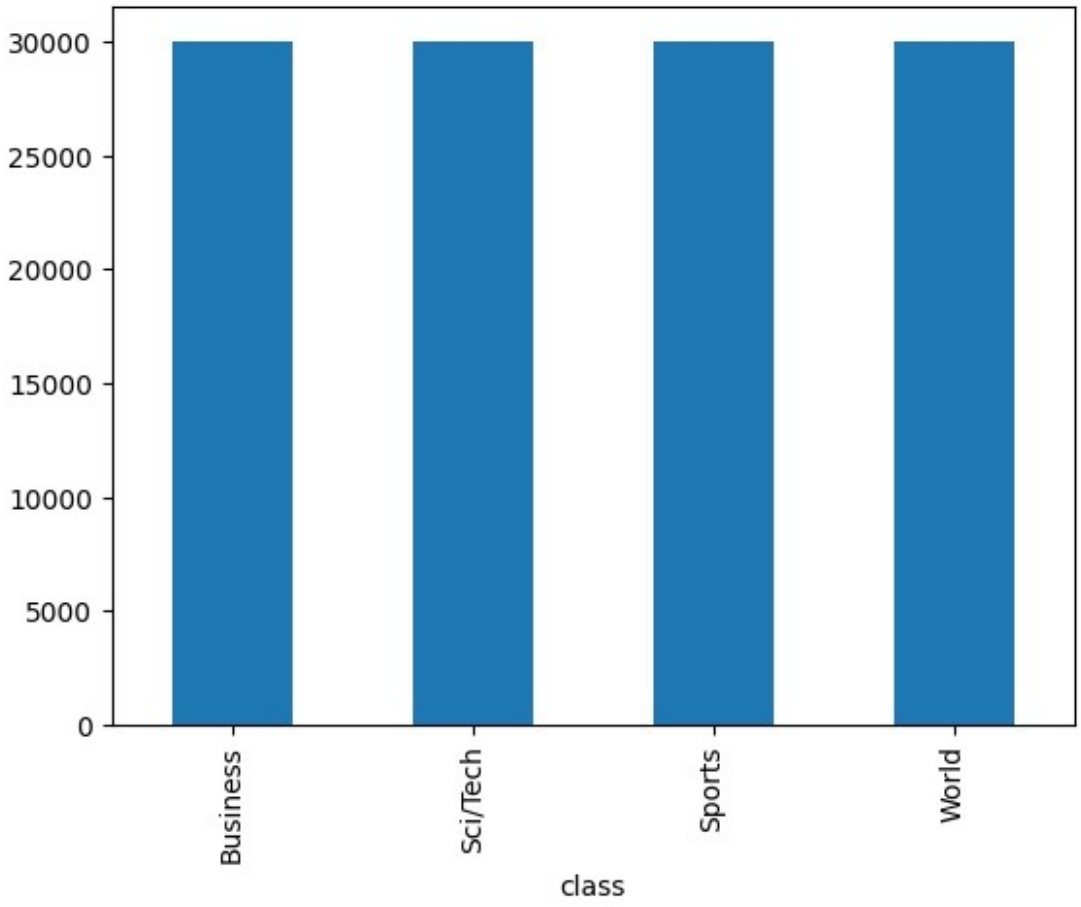
```
1       Carlyle Looks Toward Commercial Aerospace (Reu...
2         Oil and Economy Cloud Stocks' Outlook (Reuters)
3       Iraq Halts Oil Exports from Main Southern Pipe...
4       Oil prices soar to all-time record, posing new...
...                                                   ...
119995  Pakistan's Musharraf Says Won't Quit as Army C...
119996                  Renteria signing a top-shelf deal
119997                    Saban not going to Dolphins yet
119998                                Today's NFL games
119999                     Nets get Carter from Raptors

                                                description
0       Reuters - Short-sellers, Wall Street's dwindli...
1       Reuters - Private investment firm Carlyle Grou...
2       Reuters - Soaring crude prices plus worries\ab...
3       Reuters - Authorities have halted oil export\f...
4       AFP - Tearaway world oil prices, toppling reco...
...                                                   ...
119995   KARACHI (Reuters) - Pakistani President Perve...
119996  Red Sox general manager Theo Epstein acknowled...
119997  The Miami Dolphins will put their courtship of...
119998  PITTSBURGH at NY GIANTS Time: 1:30 p.m. Line: ...
119999  INDIANAPOLIS -- All-Star Vince Carter was trad...

[120000 rows x 4 columns]
```

Let's inspect how balanced our examples are by using a bar plot.

En este chunk de código generamos una gráfica de barras para verificar si las clases se distribuyen de manera uniforme, porque cuando trabajamos con problemas de clasificación lo que no queremos es que haya un desequilibrio ya que no queremos que el modelo favorezca más una clase que otra, como vemos aqui no tenemos ese problema porque la grafica se ve equilibrada.

```
pd.value_counts(train_df['class']).plot.bar()
```

```
/tmp/ipykernel_30/1245903889.py:1: FutureWarning: pandas.value_counts
is deprecated and will be removed in a future version. Use
pd.Series(obj).value_counts() instead.
  pd.value_counts(train_df['class']).plot.bar()

<Axes: xlabel='class'>
```



The classes are evenly distributed. That's great!

However, the text contains some spurious backslashes in some parts of the text. They are meant to represent newlines in the original text. An example can be seen below, between the words "dwindling" and "band".

En este chunk de código inspeccionamos una de las descripciones en el conjunto de datos para identificar posibles problemas en el texto.Estos caracteres pueden interferir con el procesamiento de texto posterior,por lo que se busca identificarlos para limpiarlos y que el texto no reciba errores ni ruido.

```
print(train_df.loc[0, 'description'])
```

```
Reuters - Short-sellers, Wall Street's dwindling\band of ultra-cynics,
are seeing green again.
```

We will replace the backslashes with spaces on the whole column using pandas replace method.

En este chunk de código se hace una limpieza, se convierten el titulo 'title' y la descripcion 'description' en minusculas y luego se unifican al juntarse en una misma y nueva columna llamada 'text'. Se reemplaza el caracter '' por un espacio en blanco para la limpieza.

```
train_df['text'] = train_df['title'].str.lower() + " " +
train_df['description'].str.lower()
train_df['text'] = train_df['text'].str.replace('\\', ' ',
regex=False)
train_df
```

```
        class index     class  \
0                 3  Business
1                 3  Business
2                 3  Business
3                 3  Business
4                 3  Business
...             ...       ...
119995            1     World
119996            2    Sports
119997            2    Sports
119998            2    Sports
119999            2    Sports

                                                    title  \
0       Wall St. Bears Claw Back Into the Black (Reuters)
1       Carlyle Looks Toward Commercial Aerospace (Reu...
2         Oil and Economy Cloud Stocks' Outlook (Reuters)
3       Iraq Halts Oil Exports from Main Southern Pipe...
4       Oil prices soar to all-time record, posing new...
...                                                   ...
119995  Pakistan's Musharraf Says Won't Quit as Army C...
119996                  Renteria signing a top-shelf deal
119997                    Saban not going to Dolphins yet
119998                                Today's NFL games
119999                     Nets get Carter from Raptors

                                              description  \
0       Reuters - Short-sellers, Wall Street's dwindli...
1       Reuters - Private investment firm Carlyle Grou...
2       Reuters - Soaring crude prices plus worries\ab...
3       Reuters - Authorities have halted oil export\f...
4       AFP - Tearaway world oil prices, toppling reco...
...                                                   ...
119995   KARACHI (Reuters) - Pakistani President Perve...
119996  Red Sox general manager Theo Epstein acknowled...
119997  The Miami Dolphins will put their courtship of...
119998  PITTSBURGH at NY GIANTS Time: 1:30 p.m. Line: ...
119999  INDIANAPOLIS -- All-Star Vince Carter was trad...

                                                     text
0       wall st. bears claw back into the black (reute...
1       carlyle looks toward commercial aerospace (reu...
2       oil and economy cloud stocks' outlook (reuters...
3       iraq halts oil exports from main southern pipe...
```

```
4       oil prices soar to all-time record, posing new...
...                                                   ...
119995  pakistan's musharraf says won't quit as army c...
119996  renteria signing a top-shelf deal red sox gene...
119997  saban not going to dolphins yet the miami dolp...
119998  today's nfl games pittsburgh at ny giants time...
119999  nets get carter from raptors indianapolis -- a...

[120000 rows x 5 columns]
```

Now we will proceed to tokenize the title and description columns using NLTK's word_tokenize(). We will add a new column to our dataframe with the list of tokens.

En este chunk de código es donde se lleva a cabo el proceso de tokenización de texto, es importante que se haga adecuadamente ya que esta parte es fundamental en el preprocesamiento para modelos de NLP. Usamos la función word_tokenize de la libreria nltk se divide el texto de cada registro en palabras individuales (tokens) . word_tokenize va a crear una lista de tokens para cada fila en la columna text, la lsita se va a almacenar en una columna nueva llamda 'tokens'. Con la función progress_map vamos a poder visualizar el progreso que va llevando del proceso de tokenización para cada resgitro.

```
from nltk.tokenize import word_tokenize

train_df['tokens'] = train_df['text'].progress_map(word_tokenize)
train_df
```


```
{"model_id":"0e92e48c6cc04f3d90230aa226cbf79d","version_major":2,"vers
ion_minor":0}
```


```
        class index     class  \
0                 3  Business
1                 3  Business
2                 3  Business
3                 3  Business
4                 3  Business
...             ...       ...
119995            1     World
119996            2    Sports
119997            2    Sports
119998            2    Sports
119999            2    Sports

                                                    title  \
0       Wall St. Bears Claw Back Into the Black (Reuters)
1       Carlyle Looks Toward Commercial Aerospace (Reu...
2         Oil and Economy Cloud Stocks' Outlook (Reuters)
3       Iraq Halts Oil Exports from Main Southern Pipe...
4       Oil prices soar to all-time record, posing new...
...                                                   ...
119995  Pakistan's Musharraf Says Won't Quit as Army C...
```

```
119996                   Renteria signing a top-shelf deal
119997                     Saban not going to Dolphins yet
119998                                  Today's NFL games
119999                        Nets get Carter from Raptors

                                                description  \
0       Reuters - Short-sellers, Wall Street's dwindli...
1       Reuters - Private investment firm Carlyle Grou...
2       Reuters - Soaring crude prices plus worries\ab...
3       Reuters - Authorities have halted oil export\f...
4       AFP - Tearaway world oil prices, toppling reco...
...                                                   ...
119995   KARACHI (Reuters) - Pakistani President Perve...
119996  Red Sox general manager Theo Epstein acknowled...
119997  The Miami Dolphins will put their courtship of...
119998  PITTSBURGH at NY GIANTS Time: 1:30 p.m. Line: ...
119999  INDIANAPOLIS -- All-Star Vince Carter was trad...

                                                    text  \
0       wall st. bears claw back into the black (reute...
1       carlyle looks toward commercial aerospace (reu...
2       oil and economy cloud stocks' outlook (reuters...
3       iraq halts oil exports from main southern pipe...
4       oil prices soar to all-time record, posing new...
...                                                   ...
119995  pakistan's musharraf says won't quit as army c...
119996  renteria signing a top-shelf deal red sox gene...
119997  saban not going to dolphins yet the miami dolp...
119998  today's nfl games pittsburgh at ny giants time...
119999  nets get carter from raptors indianapolis -- a...

                                                   tokens
0       [wall, st., bears, claw, back, into, the, blac...
1       [carlyle, looks, toward, commercial, aerospace...
2       [oil, and, economy, cloud, stocks, ', outlook,...
3       [iraq, halts, oil, exports, from, main, southe...
4       [oil, prices, soar, to, all-time, record, ,, p...
...                                                   ...
119995  [pakistan, 's, musharraf, says, wo, n't, quit,...
119996  [renteria, signing, a, top-shelf, deal, red, s...
119997  [saban, not, going, to, dolphins, yet, the, mi...
119998  [today, 's, nfl, games, pittsburgh, at, ny, gi...
119999  [nets, get, carter, from, raptors, indianapoli...

[120000 rows x 6 columns]
```

Now we will load the GloVe word embeddings.

En este chunk de código se carga el modelo de vectores de palabras preentrenado GloVe (Global Vectors for Word Representation) usando gensim. Vamos a usar el formato

load_word2vec_format para cargar el archivo de embeddings glove.6B.300d.txt, este contiene representaciones de 300 dimensiones para 400,000 palabras. La función glove.vectors.shape sirve para que devuelva las dimensiones de la matriz de embeddings y ya con eso confirmamos que tiene 400,000 palabras con 300 características cada una y ya estos vectores nos van a permitir representar palabras en un espacio semántico para mejorar el rendimiento del modelo en tareas de NLP.

```
from gensim.models import KeyedVectors
glove = KeyedVectors.load_word2vec_format("/kaggle/input/glove-6b-
300d-txt/glove.6B.300d.txt", no_header=True)
glove.vectors.shape
```

```
(400000, 300)
```

The word embeddings have been pretrained in a different corpus, so it would be a good idea to estimate how good our tokenization matches the GloVe vocabulary.

En este chubk de código lo que se hace es analizar la cantidad de tokens desconocidos en el corpus de entrenamiento en relación con el vocabulario de GloVe. Se define la función count_unknown_words para hacer el registro de cuántas veces aparecen tokens que no están en el vocabulario de GloVe, pasando por cada token en el conjunto de datos y contando los desconocidos. Despues se calcula el número total de tokens, la cantidad de tokens desconocidos (unk_tokens), y el porcentaje de tokens desconocidos en relación al total (percent_unk) y cuantos tokens únicos desconocidos hay y con prints imprimimos un resumen de estos datos. Es importante este paso porque para el rendimiento del modelo NLP es importante identificar palabras o símbolos que no tienen representación en el modelo de GloVe-

```
from collections import Counter

def count_unknown_words(data, vocabulary):
    counter = Counter()
    for row in tqdm(data):
        counter.update(tok for tok in row if tok not in vocabulary)
    return counter

# find out how many times each unknown token occurrs in the corpus
c = count_unknown_words(train_df['tokens'], glove.key_to_index)

# find the total number of tokens in the corpus
total_tokens = train_df['tokens'].map(len).sum()

# find some statistics about occurrences of unknown tokens
unk_tokens = sum(c.values())
percent_unk = unk_tokens / total_tokens
distinct_tokens = len(list(c))

print(f'total number of tokens: {total_tokens:,}')
print(f'number of unknown tokens: {unk_tokens:,}')
print(f'number of distinct unknown tokens: {distinct_tokens:,}')
print(f'percentage of unkown tokens: {percent_unk:.2%}')
```

```
print('top 50 unknown words:')
for token, n in c.most_common(10):
    print(f'\t{n}\t{token}')
```

{"model_id":"ad5cd1d6f09c4670a298a152cac8d1e6","version_major":2,"version_minor":0}

```
total number of tokens: 5,273,096
number of unknown tokens: 66,008
number of distinct unknown tokens: 24,792
percentage of unkown tokens: 1.25%
top 50 unknown words:
     2984  /b
     2119  href=
     2117  /a
     1813  //www.investor.reuters.com/fullquote.aspx
     1813  target=/stocks/quickinfo/fullquote
     537   /p
     510   newsfactor
     471   cbs.mw
     431   color=
     417   /font
```

Glove embeddings seem to have a good coverage on this dataset -- only 1.25% of the tokens in the dataset are unknown, i.e., don't appear in the GloVe vocabulary.

Still, we will need a way to handle these unknown tokens. Our approach will be to add a new embedding to GloVe that will be used to represent them. This new embedding will be initialized as the average of all the GloVe embeddings.

We will also add another embedding, this one initialized to zeros, that will be used to pad the sequences of tokens so that they all have the same length. This will be useful when we train with mini-batches.

En este chunk de código se van a agregar dos tokens especiales al vocabulario de embeddings de GloVe: [UNK] para palabras desconocidas y [PAD] para padding. Despues se van a definir unk_tok y pad_tok como las cadenas para estos tokens especiales. Despues se inicializan sus vectores de embeddings: unk_emb se establece como el promedio de todos los vectores de GloVe (palabras desconocidas), y pad_emb se define como un vector de ceros y por ultimo se agregan estos nuevos vectores al modelo de embeddings de GloVe con add_vectors. Y ya por ultimo se imprimen los indices de unk_id y pad_id. esto es util para que el modelo maneje de forma consistente palabras desconocidas y padding en la entrada.

```
# string values corresponding to the new embeddings
unk_tok = '[UNK]'
pad_tok = '[PAD]'

# initialize the new embedding values
unk_emb = glove.vectors.mean(axis=0)
pad_emb = np.zeros(300)
```

```
# add new embeddings to glove
glove.add_vectors([unk_tok, pad_tok], [unk_emb, pad_emb])

# get token ids corresponding to the new embeddings
unk_id = glove.key_to_index[unk_tok]
pad_id = glove.key_to_index[pad_tok]

unk_id, pad_id
```

```
(400000, 400001)
```

En este chunk de código se divide el conjunto de datos original en un 80% para entrenamiento o se el train_df y un 20% para validación , o sea dev_df usando train_test_split. Despues se restablecen los índices en los dos DataFrames para facilitar su uso. Esta división permite evaluar el rendimiento del modelo en datos de validación.

```
from sklearn.model_selection import train_test_split

train_df, dev_df = train_test_split(train_df, train_size=0.8)
train_df.reset_index(inplace=True)
dev_df.reset_index(inplace=True)
```

We will now add a new column to our dataframe that will contain the padded sequences of token ids.

En este chunk de código se crea el vocabulario a partir de los tokens en el conjunto de entrenamiento (train_df) , solamente se retienen los que aparecen más de 10 veces (threshold=10). Primero contamos la frecuencia de cada token y despues se filtran los tokens que si cumplen el umbral, luego estos tokens se almacenan en un conjunto (vocabulary) para eliminar duplicados. Y por último se imprime el tamaño del vocabulario (vocabulary size) , que es de 17,441 tokens.

```
threshold = 10
tokens = train_df['tokens'].explode().value_counts()
vocabulary = set(tokens[tokens > threshold].index.tolist())
print(f'vocabulary size: {len(vocabulary):,}')
```

```
vocabulary size: 17,441
```

```
# find the length of the longest list of tokens
max_tokens = train_df['tokens'].map(len).max()

# return unk_id for infrequent tokens too
def get_id(tok):
    if tok in vocabulary:
        return glove.key_to_index.get(tok, unk_id)
    else:
        return unk_id
```

```
# function that gets a list of tokens and returns a list of token ids,
# with padding added accordingly
def token_ids(tokens):
    tok_ids = [get_id(tok) for tok in tokens]
    pad_len = max_tokens - len(tok_ids)
    return tok_ids + [pad_id] * pad_len

# add new column to the dataframe
train_df['token ids'] = train_df['tokens'].progress_map(token_ids)
train_df
```

{"model_id":"0b3991df7d6841aa94ccafb520597536","version_major":2,"version_minor":0}

	index	class index	class	\
0	9116	1	World	
1	99831	3	Business	
2	10663	3	Business	
3	73175	4	Sci/Tech	
4	104494	4	Sci/Tech	
...	...	...	...	
95995	89460	1	World	
95996	60620	1	World	
95997	34086	1	World	
95998	58067	1	World	
95999	92975	4	Sci/Tech	

	title	\
0	Najaf's Residents Feel Trapped in Battle (AP)	
1	U.S. FDA Adds Restrictions to Acne Drug	
2	Smithfield Foods Profit More Than Doubles	
3	PluggedIn: The OQO Is Not Just Another Handhel...	
4	IBM invigorates LTO tape storage	
...	...	
95995	Bush, Blair See Hope for Palestinian State (AP)	
95996	Ex-Soldiers Vow to Bring Order to Haiti Capital	
95997	Musharraf says U.S. must address root of terro...	
95998	Nuclear materials #39;vanish #39; in Iraq	
95999	In Brief: Bowstreet unveils pre-packaged porta...	

	description	\
0	AP - For nearly three weeks, Amer al-Jamali ha...	
1	WASHINGTON (Reuters) - Roche's acne drug Accu...	
2	Smithfield Foods Inc. (SFD.N: Quote, Profile, ...	
3	SAN FRANCISCO (Reuters) - A full-fledged Wind...	
4	LTO (linear tape open)-based drives are invigo...	
...	...	
95995	AP - As Yasser Arafat was buried, President Bu...	
95996	Ex-soldiers who helped topple former President...	
95997	Reuters - The United States could lose its war...	

```
95998  Equipment and materials that could be used to ...
95999  Bowstreet this week launched its Enterprise Po...

                                                    text  \
0      najaf's residents feel trapped in battle (ap) ...
1      u.s. fda adds restrictions to acne drug  washi...
2      smithfield foods profit more than doubles smit...
3      pluggedin: the oqo is not just another handhel...
4      ibm invigorates lto tape storage lto (linear t...
...                                                  ...
95995  bush, blair see hope for palestinian state (ap...
95996  ex-soldiers vow to bring order to haiti capita...
95997  musharraf says u.s. must address root of terro...
95998  nuclear materials  #39;vanish #39; in iraq equ...
95999  in brief: bowstreet unveils pre-packaged porta...

                                                  tokens  \
0      [najaf, 's, residents, feel, trapped, in, batt...
1      [u.s., fda, adds, restrictions, to, acne, drug...
2      [smithfield, foods, profit, more, than, double...
3      [pluggedin, :, the, oqo, is, not, just, anothe...
4      [ibm, invigorates, lto, tape, storage, lto, (,...
...                                                  ...
95995  [bush, ,, blair, see, hope, for, palestinian, ...
95996  [ex-soldiers, vow, to, bring, order, to, haiti...
95997  [musharraf, says, u.s., must, address, root, o...
95998  [nuclear, materials, #, 39, ;, vanish, #, 39, ...
95999  [in, brief, :, bowstreet, unveils, pre-package...

                                                token ids
0      [10709, 9, 1048, 998, 4799, 6, 903, 23, 1582, ...
1      [99, 5584, 2144, 3252, 4, 400000, 780, 289, 23...
2      [34026, 5008, 1269, 56, 73, 4229, 34026, 5008,...
3      [400000, 45, 0, 293697, 14, 36, 120, 170, 2099...
4      [5199, 400000, 400000, 4143, 4418, 400000, 23,...
...                                                  ...
95995  [272, 1, 2356, 253, 824, 10, 463, 92, 23, 1582...
95996  [223970, 12887, 4, 938, 460, 4, 3836, 351, 223...
95997  [3820, 210, 99, 390, 1476, 5440, 3, 1291, 23, ...
95998  [490, 2176, 2749, 3403, 89, 25736, 2749, 3403,...
95999  [6, 2461, 45, 400000, 20465, 400000, 12174, 83...

[96000 rows x 8 columns]
```

En este chunk código se sigue el mismo proceso de conversión y padding a los datos de validación (dev_df). Primerose tiene que calcular la longitud máxima de tokens en el conjunto de validación (max_tokens), despues usamos la función token_ids para convertir cada lista de tokens en una lista de IDs, agregando padding hasta alcanzar la longitud máxima. Y el resultado se va a guarda en una nueva columna token ids en dev_df y ya con eso podemos trabajar con el conjunto de validación.

```
max_tokens = dev_df['tokens'].map(len).max()
dev_df['token ids'] = dev_df['tokens'].progress_map(token_ids)
dev_df
```

{"model_id":"3e3b92d08e614824ad5ce09f299b692f","version_major":2,"version_minor":0}

	index	class index	class	\
0	60974	1	World	
1	50391	4	Sci/Tech	
2	9307	3	Business	
3	35221	3	Business	
4	40081	1	World	
...	...	...	...	
23995	49572	1	World	
23996	40409	4	Sci/Tech	
23997	70470	2	Sports	
23998	7941	4	Sci/Tech	
23999	42303	1	World	

	title	\
0	Sharon Accepts Plan to Reduce Gaza Army Operat...	
1	Internet Key Battleground in Wildlife Crime Fight	
2	July Durable Good Orders Rise 1.7 Percent	
3	Growing Signs of a Slowing on Wall Street	
4	The New Faces of Reality TV	
...	...	
23995	Iraqi Kidnappers Release 2 Indonesian Women	
23996	Big Wi-Fi Project for Philadelphia	
23997	Owen scores again	
23998	US Online Retail Sales Expected To Double In S...	
23999	Egyptian holding company says it has heard fou...	

	description	\
0	Israeli Prime Minister Ariel Sharon accepted a...	
1	Why trawl through a sweaty illegal\wildlife ma...	
2	America's factories saw orders for costly manu...	
3	all Street #39;s earnings growth, fueled by tw...	
4	The introduction of children to the genre was ...	
...	...	
23995	Two Indonesian women held hostage for several ...	
23996	What would Benjamin Franklin say? Philadelphia...	
23997	Michael Owen scored the winner for Real Madrid...	
23998	Online retail sales in the US are expected to ...	
23999	Egypt said Tuesday that Iraqi kidnappers had f...	

	text	\
0	sharon accepts plan to reduce gaza army operat...	
1	internet key battleground in wildlife crime fi...	
2	july durable good orders rise 1.7 percent amer...	

```
3      growing signs of a slowing on wall street all ...
4      the new faces of reality tv the introduction o...
...                                                  ...
23995  iraqi kidnappers release 2 indonesian women tw...
23996  big wi-fi project for philadelphia what would ...
23997  owen scores again michael owen scored the winn...
23998  us online retail sales expected to double in s...
23999  egyptian holding company says it has heard fou...

                                                  tokens  \
0      [sharon, accepts, plan, to, reduce, gaza, army...
1      [internet, key, battleground, in, wildlife, cr...
2      [july, durable, good, orders, rise, 1.7, perce...
3      [growing, signs, of, a, slowing, on, wall, str...
4      [the, new, faces, of, reality, tv, the, introd...
...                                                  ...
23995  [iraqi, kidnappers, release, 2, indonesian, wo...
23996  [big, wi-fi, project, for, philadelphia, what,...
23997  [owen, scores, again, michael, owen, scored, t...
23998  [us, online, retail, sales, expected, to, doub...
23999  [egyptian, holding, company, says, it, has, he...

                                               token ids
0      [2548, 9889, 394, 4, 1680, 1166, 330, 957, 1, ...
1      [925, 638, 14944, 6, 4446, 1340, 838, 738, 400...
2      [375, 10699, 219, 1949, 1027, 6262, 72, 453, 9...
3      [988, 1867, 3, 7, 6515, 13, 1015, 491, 64, 491...
4      [0, 50, 1919, 3, 2532, 816, 0, 4344, 3, 271, 4...
...                                                  ...
23995  [710, 9349, 713, 232, 2656, 266, 55, 2656, 266...
23996  [365, 39300, 716, 10, 2201, 102, 54, 4067, 503...
23997  [7116, 2776, 378, 785, 7116, 878, 0, 1364, 10,...
23998  [95, 1292, 2645, 526, 287, 4, 1278, 6, 228, 82...
23999  [2434, 1383, 128, 210, 20, 31, 1435, 133, 2434...

[24000 rows x 8 columns]
```

Now we will get a numpy 2-dimensional array corresponding to the token ids, and a 1-dimensional array with the gold classes. Note that the classes are one-based (i.e., they start at one), but we need them to be zero-based, so we need to subtract one from this array.

En este chunk de código definimos una clase personalizada de PyTorch Dataset que se llama MyDataset, que es para hacer más facil la gestión de los datos para el modelo. La clase toma como entrada x (características) e y (etiquetas).

En el método init, lo que se hace es que se inicializan los datos de entrada (self.x) y las etiquetas (self.y). y luego len va a devolver el número de muestras en el conjunto de datos, esto le va a permitir a PyTorch saber cuántos ejemplos hay. getitem lo que hace es tomar un índice (index) y devuelve la muestra correspondiente de x e y como tensores de PyTorch. Esta estructura deja

usar MyDataset con DataLoaders de PyTorch, lo cual es para facilitar el manejo de lotes y la iteración sobre el conjunto de datos.

```
from torch.utils.data import Dataset

class MyDataset(Dataset):
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __len__(self):
        return len(self.y)

    def __getitem__(self, index):
        x = torch.tensor(self.x[index])
        y = torch.tensor(self.y[index])
        return x, y
```

Next, we construct our PyTorch model, which is a feed-forward neural network with two layers:

Este chunk de código que es más complejo y largo que los demás va a definir una clase de PyTorch llamada Model la cual es una red neuronal diseñada para clasificación utilizando embeddings preentrenados.

En el método init, el modelo va a tomar como parámetros:

vectors: los embeddings preentrenados, como por ejemplo GloVe, que se va a convertir en un tensor si es que aún no lo son. pad_id: es el índice de padding, se va a usar para ignorar tokens de padding en los cálculos. hidden_dim, output_dim: son dimensiones de la capa oculta y la capa de salida. dropout: la tasa de dropout para regularización.

El modelo cuenta con una capa de embeddings (self.embs) inicializada con los vectores preentrenados, y una secuencia de capas lineales (self.layers) con funciones de activación y dropout para mejorar la capacidad de generalización.

Y en el método forward:

Se va a calcular una máscara para lograr identificar tokens que no son padding o sea (not_padding). Tambien se calcula la longitud real de cada secuencia (excluyendo padding). Se obtiene el embedding de cada token y se calcula el promedio a nivel de oración (se logra hacer dividiendo la suma de embeddings entre la longitud de la oración). Y el promedio que resulte va a pasar por las capas feedforward para obtener la predicción final. Este modelo aplica embeddings a cada token y obtiene representaciones de oraciones, ignorando el padding y aplicando regularización con dropout.

```
from torch import nn
import torch.nn.functional as F

class Model(nn.Module):
    def __init__(self, vectors, pad_id, hidden_dim, output_dim,
dropout):
```

```
        super().__init__()
        # embeddings must be a tensor
        if not torch.is_tensor(vectors):
            vectors = torch.tensor(vectors)
        # keep padding id
        self.padding_idx = pad_id
        # embedding layer
        self.embs = nn.Embedding.from_pretrained(vectors,
padding_idx=pad_id)
        # feedforward layers
        self.layers = nn.Sequential(
            nn.Dropout(dropout),
            nn.Linear(vectors.shape[1], hidden_dim),
            nn.ReLU(),
            nn.Dropout(dropout),
            nn.Linear(hidden_dim, output_dim),
        )

    def forward(self, x):
        # get boolean array with padding elements set to false
        not_padding = torch.isin(x, self.padding_idx, invert=True)
        # get lengths of examples (excluding padding)
        lengths = torch.count_nonzero(not_padding, axis=1)
        # get embeddings
        x = self.embs(x)
        # calculate means
        x = x.sum(dim=1) / lengths.unsqueeze(dim=1)
        # pass to rest of the model
        output = self.layers(x)
        # calculate softmax if we're not in training mode
        #if not self.training:
        #    output = F.softmax(output, dim=1)
        return output
```

Next, we implement the training procedure. We compute the loss and accuracy on the development partition after each epoch.

En este otrochunk de código más largo lo que se hace es que se entrena el modelo y evalúa su rendimiento en los conjuntos de entrenamiento y validación. Aqui tambien se van a definir los hiperparámetros clave como la tasa de aprendizaje, el tamaño del lote, el número de épocas, las dimensiones de la capa oculta y dropout y despues se inicializan el modelo, la función de pérdida, el optimizador y los DataLoaders para los datos de entrenamiento (train_dl) y validación (dev_dl).

EN el entrenamiento, por cada época (ephoc) el modelo se va a entrenar (model.train()) en lotes (batch). Para cada lote, se pasan los datos al dispositivo adecuado en este caso es el GPU , y lo que se hace es calcular las predicciones y la pérdida, se retropropagan los gradientes, y se actualizan los parámetros del modelo. Y ya que finalice cada época, se calcula la pérdida promedio (train_loss) y la precisión (train_acc) en el conjunto de entrenamiento. Despues se necesita evaluar el modelo (model.eval()) en el conjunto de validación sin calcular gradientes. La

pérdida (dev_loss) y la precisión (dev_acc) se registran para cada época. Estos resultados son muy importantes ya que permiten monitorear el desempeño y poder ajustar el modelo si es necesario.

```
from torch import optim
from torch.utils.data import DataLoader
from sklearn.metrics import accuracy_score

# hyperparameters
lr = 1e-3
weight_decay = 0
batch_size = 500
shuffle = True
n_epochs = 5
hidden_dim = 50
output_dim = len(labels)
dropout = 0.1
vectors = glove.vectors

# initialize the model, loss function, optimizer, and data-loader
model = Model(vectors, pad_id, hidden_dim, output_dim,
dropout).to(device)
loss_func = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=lr,
weight_decay=weight_decay)
train_ds = MyDataset(train_df['token ids'], train_df['class index'] -
1)
train_dl = DataLoader(train_ds, batch_size=batch_size,
shuffle=shuffle)
dev_ds = MyDataset(dev_df['token ids'], dev_df['class index'] - 1)
dev_dl = DataLoader(dev_ds, batch_size=batch_size, shuffle=shuffle)

train_loss = []
train_acc = []

dev_loss = []
dev_acc = []

# train the model
for epoch in range(n_epochs):
    losses = []
    gold = []
    pred = []
    model.train()
    for X, y_true in tqdm(train_dl, desc=f'epoch {epoch+1} (train)'):
        # clear gradients
        model.zero_grad()
        # send batch to right device
        X = X.to(device)
        y_true = y_true.to(device)
```

```
        # predict label scores
        y_pred = model(X)
        # compute loss
        loss = loss_func(y_pred, y_true)
        # accumulate for plotting
        losses.append(loss.detach().cpu().item())
        gold.append(y_true.detach().cpu().numpy())
        pred.append(np.argmax(y_pred.detach().cpu().numpy(), axis=1))
        # backpropagate
        loss.backward()
        # optimize model parameters
        optimizer.step()
    train_loss.append(np.mean(losses))
    train_acc.append(accuracy_score(np.concatenate(gold),
np.concatenate(pred)))

    model.eval()
    with torch.no_grad():
        losses = []
        gold = []
        pred = []
        for X, y_true in tqdm(dev_dl, desc=f'epoch {epoch+1} (dev)'):
            X = X.to(device)
            y_true = y_true.to(device)
            y_pred = model(X)
            loss = loss_func(y_pred, y_true)
            losses.append(loss.cpu().item())
            gold.append(y_true.cpu().numpy())
            pred.append(np.argmax(y_pred.cpu().numpy(), axis=1))
        dev_loss.append(np.mean(losses))
        dev_acc.append(accuracy_score(np.concatenate(gold),
np.concatenate(pred)))
```

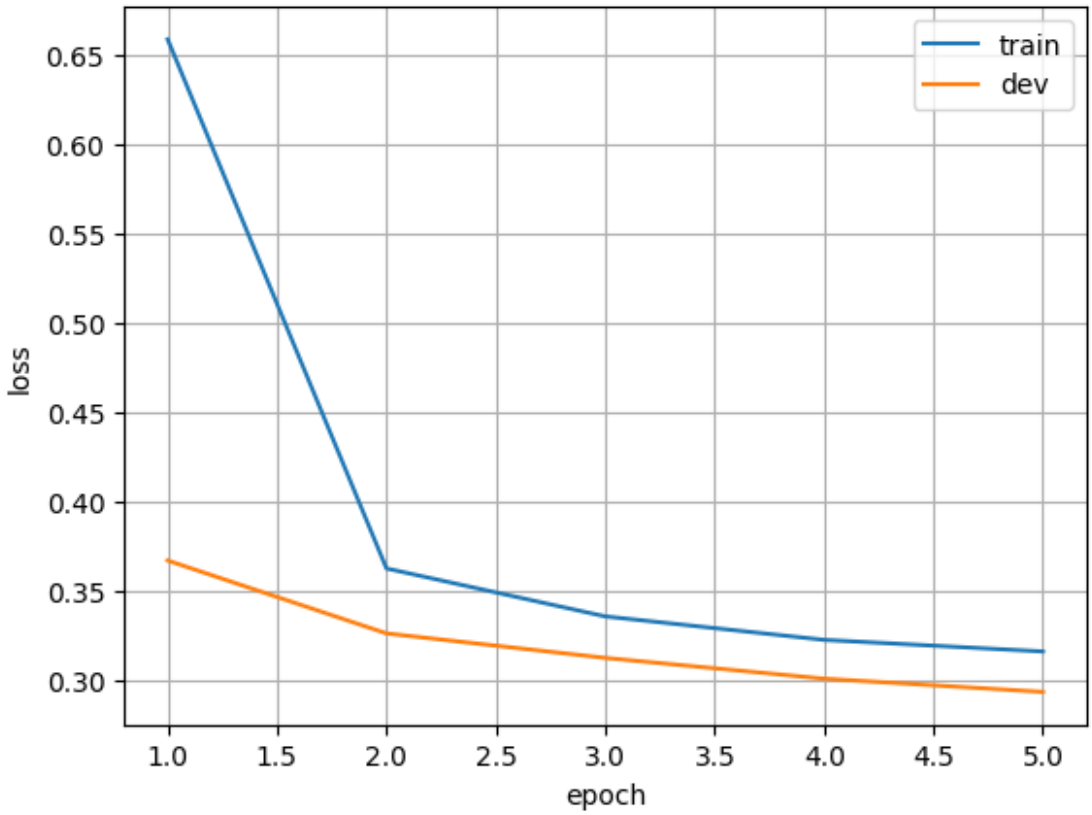

```
{"model_id":"3e9d60e97ff0496e9a484fa57e04fd3b","version_major":2,"vers
ion_minor":0}

{"model_id":"a0595677b29c49928150de924c28fe12","version_major":2,"vers
ion_minor":0}

{"model_id":"889ed61726c7403ca61075a55861b12d","version_major":2,"vers
ion_minor":0}

{"model_id":"c78dbc3f62e04fcab2670029b58d3ed4","version_major":2,"vers
ion_minor":0}

{"model_id":"df0e571126e147d0be9860a296f01ce9","version_major":2,"vers
ion_minor":0}

{"model_id":"f6bce3a72ccd4cb7987511b6a9208302","version_major":2,"vers
ion_minor":0}
```

```
{"model_id":"c05a0b252490455095146868943 82ff6","version_major":2,"vers
ion_minor":0}

{"model_id":"e578fd7253d14b2c98f85ac4b66503b1","version_major":2,"vers
ion_minor":0}

{"model_id":"34b66018fdd64b9c82febbc2d94c855c","version_major":2,"vers
ion_minor":0}

{"model_id":"1ad59eccfb144a7cadba2ba635a8a12c","version_major":2,"vers
ion_minor":0}
```


Let's plot the loss and accuracy on dev:

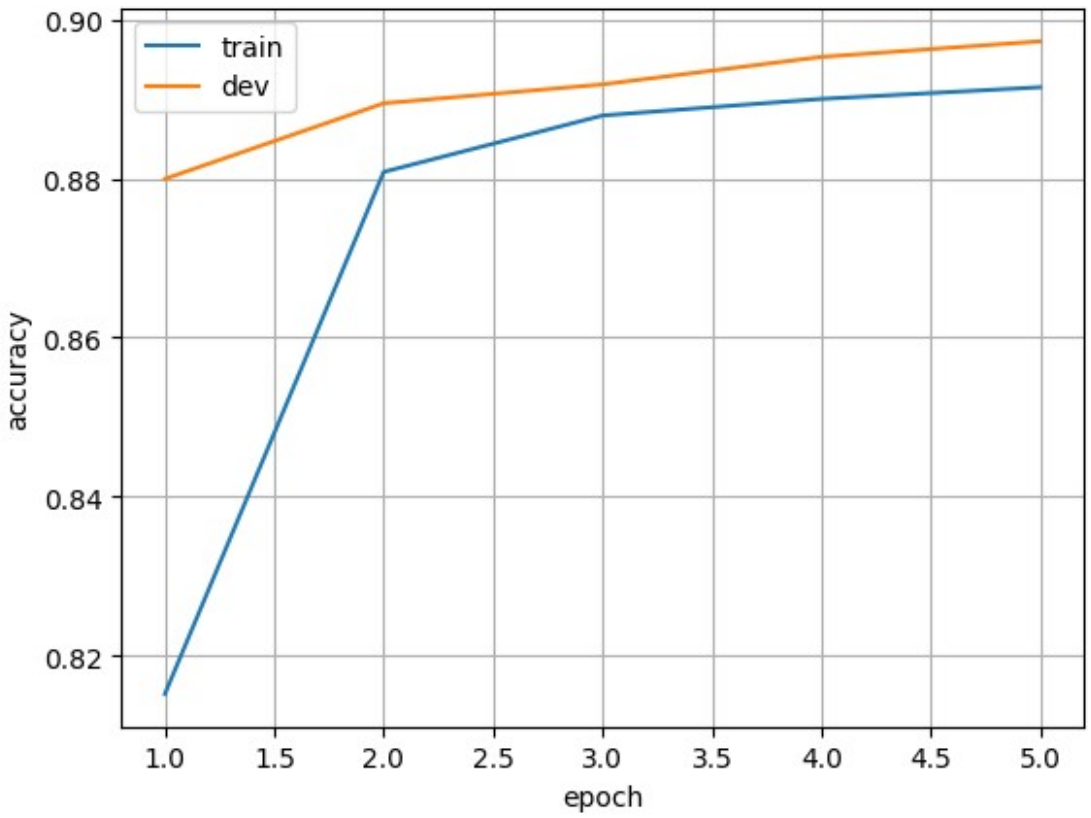
Este chunk de código grafica la pérdida de entrenamiento y validación a lo largo de las épocas y muestra cómo es que cambia la pérdida en ambos conjuntos, ayudando a identificar el ajuste del modelo.

```
import matplotlib.pyplot as plt
%matplotlib inline

x = np.arange(n_epochs) + 1

plt.plot(x, train_loss)
plt.plot(x, dev_loss)
plt.legend(['train', 'dev'])
plt.xlabel('epoch')
plt.ylabel('loss')
plt.grid(True)
```



En este chunk de código se grafica la precisión (accuracy) en los conjuntos de entrenamiento y validación a lo largo de las épocas, lo cual permite visualizar cómo mejora la precisión en ambos conjuntos y poder verificar si el modelo está generalizando bien o presenta problemas como sobreajuste.

```
plt.plot(x, train_acc)
plt.plot(x, dev_acc)
plt.legend(['train', 'dev'])
plt.xlabel('epoch')
plt.ylabel('accuracy')
plt.grid(True)
```



Next, we evaluate on the testing partition:

En este chunk de código se preprocesa el conjunto de prueba (test_df) para que coincida con los datos de entrenamiento y validación. Se cargan los datos, combina y limpia el texto, tokeniza, y convierte los tokens en IDs con padding basado en la longitud máxima del conjunto de validación y esto es para preparar el conjunto de prueba para la evaluación del modelo.

```
# repeat all preprocessing done above, this time on the test set


test_df =
pd.read_csv('/kaggle/input/ag-news-classification-dataset/test.csv',
header=None)
test_df.columns = ['class index', 'title', 'description']
test_df['text'] = test_df['title'].str.lower() + " " +
test_df['description'].str.lower()
test_df['text'] = test_df['text'].str.replace('\\', ' ', regex=False)
test_df['tokens'] = test_df['text'].progress_map(word_tokenize)
max_tokens = dev_df['tokens'].map(len).max()
test_df['token ids'] = test_df['tokens'].progress_map(token_ids)
```

{"model_id":"d40067bb76e24a2c932897c266e04338","version_major":2,"version_minor":0}

{"model_id":"f5c35a0f65fb41b1a49005a8072bf15b","version_major":2,"version_minor":0}

Se convierte class index a numérico y se eliminan las filas no convertibles, reiniciando el índice. Y ya se tiene test_df sin valores faltantes para la evaluación.

```
test_df['class index'] = pd.to_numeric(test_df['class index'], errors='coerce')
test_df = test_df.dropna(subset=['class index']).reset_index(drop=True)
```

Y ya con este ultimo chunk de código se evalúa el modelo en el conjunto de prueba, con DataLoader se hacen las predicciones sin gradientes.Despues classification_report compara las etiquetas reales con las predicciones, mostrandonos: precisión, recall y F1-score por clase, resumiendo el rendimiento del modelo.

```
from sklearn.metrics import classification_report

# set model to evaluation mode
model.eval()


dataset = MyDataset(test_df['token ids'], test_df['class index'] - 1)
data_loader = DataLoader(dataset, batch_size=batch_size)
y_pred = []

# don't store gradients
with torch.no_grad():
    for X, _ in tqdm(data_loader):
        X = X.to(device)
        # predict one class per example
        y = torch.argmax(model(X), dim=1)
        # convert tensor to numpy array (sending it back to the cpu if needed)
        y_pred.append(y.cpu().numpy())
        # print results
    print(classification_report(dataset.y, np.concatenate(y_pred), target_names=labels))
```

{"model_id":"ccd399fb663a4e258e48ea299e37eda1","version_major":2,"version_minor":0}

```
              precision    recall  f1-score   support

       World       0.92      0.88      0.90      1900
      Sports       0.95      0.97      0.96      1900
    Business       0.85      0.85      0.85      1900
```

```
    Sci/Tech       0.86      0.88      0.87      1900

    accuracy                           0.90      7600
   macro avg       0.90      0.90      0.90      7600
weighted avg       0.90      0.90      0.90      7600
```

El modelo logra un 90% de precisión general, con buenos resultados en todas las clases, en especial en "Sports" y "World".