

✓ Multiclass Text Classification with

Feed-forward Neural Networks and Word Embeddings

First, we will do some initialization.

Para iniciar importamos librerías y se configura un dispositivo GPU si está disponible. También se establece una semilla aleatoria para reproducibilidad.

```
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: cpu
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.

Ahora se carga y estructura el dataset de entrenamiento `train.csv`, asignando nombres a las columnas.

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

	class	index	title	description
0	3	Wall St. Bears Claw Back Into the Black (Reuters)	Reuters - Short-sellers, Wall Street's dwindli...	
1	3	Carlyle Looks Toward Commercial Aerospace (Reu...	Reuters - Private investment firm Carlyle Grou...	
2	3	Oil and Economy Cloud Stocks' Outlook (Reuters)	Reuters - Soaring crude prices plus worries\ab...	
3	3	Iraq Halts Oil Exports from Main Southern Pipe...	Reuters - Authorities have halted oil exportf...	
4	3	Oil prices soar to all-time record, posing new...	AFP - Tearaway world oil prices, toppling reco...	
...
119995	1	Pakistan's Musharraf Says Won't Quit as Army C...	KARACHI (Reuters) - Pakistani President Perve...	
119996	2	Renteria signing a top-shelf deal	Red Sox general manager Theo Epstein acknowl...	
119997	2	Saban not going to Dolphins yet	The Miami Dolphins will put their courtship of...	
119998	2	Today's NFL games	PITTSBURGH at NY GIANTS Time: 1:30 p.m. Line: ...	
119999	2	Nets get Carter from Raptors	INDIANAPOLIS -- All-Star Vince Carter was trad...	

120000 rows × 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.

Se cargan las etiquetas de las clases desde un archivo .txt y se mapean en el dataframe.

```
labels = open('/kaggle/input/ag-news/ag_news_csv/classes.txt').read().splitlines()
classes = train_df['class index'].map(lambda i: labels[i-1])
train_df.insert(1, 'class', classes)
train_df
```



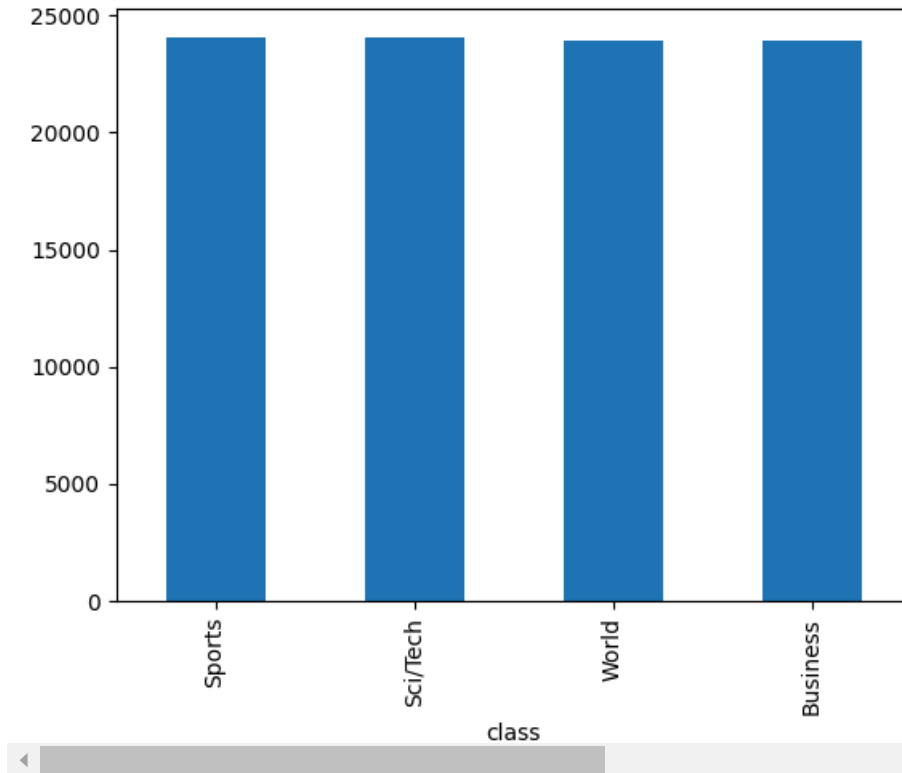
	class index	class	title	description
0	3	Business	Wall St. Bears Claw Back Into the Black (Reuters)	Reuters - Short-sellers, Wall Street's dwindli...
1	3	Business	Carlyle Looks Toward Commercial Aerospace (Reu...	Reuters - Private investment firm Carlyle Grou...
2	3	Business	Oil and Economy Cloud Stocks' Outlook (Reuters)	Reuters - Soaring crude prices plus worries\ab...
3	3	Business	Iraq Halts Oil Exports from Main Southern Pipe...	Reuters - Authorities have halted oil exportf...
4	3	Business	Oil prices soar to all-time record, posing new...	AFP - Tearaway world oil prices, toppling reco...
...
119995	1	World	Pakistan's Musharraf Says Won't Quit as Army C...	KARACHI (Reuters) - Pakistani President Perve...
119996	2	Sports	Renteria signing a top-shelf deal	Red Sox general manager Theo Epstein acknowl...

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

Realizamos una muestra aleatoria para conservar el 80% de los datos y no se acabe la memoria. También se grafica la distribución de clases en el dataset de entrenamiento.

```
train_df=train_df.sample(frac=0.8, random_state=42)
pd.value_counts(train_df['class']).plot.bar()
```

⚠ /tmp/ipykernel_30/1641164020.py:2: FutureWarning: pandas.value_counts is deprecated and will be removed in a future version. Use pd.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".

Se imprime la descripción del primer registro.


```
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.

Se convierte el texto a minúsculas y se reemplazan los caracteres \ para unificar el formato.

```
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	title	description	text
71787	3	Business	BBC set for major shake-up, claims newspaper	London - The British Broadcasting Corporation,...	bbc set for major shake-up, claims newspaper l...
67218	3	Business	Marsh averts cash crunch	Embattled insurance broker #39;s banks agree t...	marsh averts cash crunch embattled insurance b...
54066	2	Sports	Jeter, Yankees Look to Take Control (AP)	AP - Derek Jeter turned a season that started ...	jeter, yankees look to take control (ap) ap - ...
7168	4	Sci/Tech	Flying the Sun to Safety	When the Genesis capsule comes back to Earth w...	flying the sun to safety when the genesis caps...
29618	3	Business	Stocks Seen Flat as Nortel and Oil Weigh	NEW YORK (Reuters) - U.S. stocks were set to ...	stocks seen flat as nortel and oil weigh new ...
...
59228	4	Sci/Tech	Investors Flock to Web Networking Sites	Internet whiz kids Marc Andreessen, Josh Kopel...	investors flock to web networking sites intern...
64447	3	Business	Samsung Electric Quarterly	Samsung Electronics Co. Ltd.	samsung electric quarterly profit

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.

Usamos word_tokenize de nltk para dividir el texto en tokens.

```
from nltk.tokenize import word_tokenize
```

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

0%| 0/96000 [00:00<?, ?it/s]

	class index	class	title	description	text	tokens
71787	3	Business	BBC set for major shake-up, claims newspaper	London - The British Broadcasting Corporation,...	bbc set for major shake-up, claims newspaper l...	[bbc, set, for, major, shake-up, ,, claims, ne...
67218	3	Business	Marsh averts cash crunch	Embattled insurance broker #39;s banks agree t...	marsh averts cash crunch embattled insurance b...	[marsh, averts, cash, crunch, embattled, insur...
54066	2	Sports	Jeter, Yankees Look to Take Control (AP)	AP - Derek Jeter turned a season that started ...	jeter, yankees look to take control (ap) ap - ...	[jeter, ,, yankees, look, to, take, control, (...]
7168	4	Sci/Tech	Flying the Sun to Safety	When the Genesis capsule comes back to Earth w...	flying the sun to safety when the genesis caps...	[flying, the, sun, to, safety, when, the, gene...
29618	3	Business	Stocks Seen Flat as Nortel and Oil Weigh	NEW YORK (Reuters) - U.S. stocks were set to ...	stocks seen flat as nortel and oil weigh new ...	[stocks, seen, flat, as, nortel, and, oil, wei...
...

Now we will load the GloVe word embeddings.

Cargamos un modelo de embeddings GloVe preentrenado.

```
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.

A continuación se calcula la frecuencia de palabras desconocidas en el vocabulario GloVe y se analiza su distribución en el corpus.

```
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 occurs 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 unknown tokens: {percent_unk:.2%}')
print('top 50 unknown words:')
for token, n in c.most_common(10):
    print(f'\t{n}\t{token}')
```

```
0%|          | 0/96000 [00:00<?, ?it/s]
total number of tokens: 4,218,415
number of unknown tokens: 52,899
number of distinct unknown tokens: 20,979
percentage of unknown tokens: 1.25%
top 50 unknown words:
    2379    /b
    1708    href=
    1707    /a
    1461    //www.investor.reuters.com/fullquote.aspx
    1461    target=/stocks/quickinfo/fullquote
    450     /p
    396    newsfactor
    380    cbs.mw
    344    color=
    333    f--
```

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.

Se añade embeddings personalizados [UNK] y [PAD] para palabras desconocidas y padding respectivamente.

```
# 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)
```

Ahora se divide el conjunto de entrenamiento en subconjuntos de entrenamiento y 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.

Filtramos el vocabulario para incluir solo las palabras con una frecuencia mayor al umbral de 10.

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

→ vocabulary size: 15,451
```

Se convierte tokens a IDs, aplicando padding si es necesario para mantener longitudes consistentes.

```
# 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
```

0%| | 0/76800 [00:00<?, ?it/s]

	index	class index	class	title	description	text	tokens	token ids
0	41480	3	Business	Unrest forces oil prices higher	Oil futures have jumped to their highest closi...	unrest forces oil prices higher oil futures ha...	[unrest, forces, oil, prices, higher, oil, fut...	[4615, 340, 316, 468, 609, 316, 3081, 33, 3450...
1	112119	4	Sci/Tech	Old News.... REALLY Old News!	The video archives of Pathe News are online, c...	old news.... really old news! the video archiv...	[old, news, ..., , really, old, news, !, the,...	[167, 172, 434, 2, 588, 167, 172, 805, 0, 974,...
2	75220	2	Sports	Ace in the Hole	General Manager Theo Epstein said the Red Sox ...	ace in the hole general manager theo epstein s...	[ace, in, the, hole, general, manager, theo, e...	[7588, 6, 0, 2924, 216, 865, 15599, 17434, 16,...
3	111911	2	Sports	UNDATED: 14 points.	Tiffany Porter-Talbert scored 24 points, and W...	undated: 14 points. tiffany porter-talbert sco...	[undated, :, 14, points, ,, tiffany, porter-ta...	[5833, 45, 657, 226, 2, 15956, 400000, 878, 79...
4	80697	2	Sports	Flatley, Rogers on bench for Australia for rug...	Back from injury, Elton Flatley and Mat Rogers...	flatley, rogers on bench for australia for rug...	[flatley, ,, rogers, on, bench, for, australia...	[400000, 1, 5638, 13, 4530, 10, 603, 10, 2707,...
...
76795	110136	2	Sports	Gerrard aiming high	Steven Gerrard insists he #39;ll not accept q...	gerrard aiming high steven gerrard insists he ...	[gerrard, aiming, high, steven, gerrard, insis...	[15773, 7584, 152, 4411, 15773, 4971, 18, 2749...

Se realiza el mismo procedimiento de tokenización y padding en 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
```

0% | 0/19200 [00:00<?, ?it/s]

	index	class index	class	title	description	text	tokens	token ids
0	96457	1	World	House G.O.P. Leader Hails Ethics Panel's Rebuk...	Tom DeLay of Texas claimed vindication today a...	house g.o.p. leader hails ethics panel's rebuk...	[house, g.o.p., leader, hails, ethics, panel...	[166, 400000, 2, 329, 15244, 5321, 1908, 9, 19...
1	65284	2	Sports	Pittsburgh Steelers Notes	Bill Cowher is no longer 0-for-Texas. He beat ...	pittsburgh steelers notes bill cowher is no lo...	[pittsburgh, steelers, notes, bill, cowher, is...	[3576, 9841, 2142, 480, 400000, 14, 84, 1078, ...
2	48958	1	World	US, Iraq control Samarra	SAMARRA, Iraq - US and Iraqi forces in Samarra...	us, iraq control samarra samarra, iraq - us an...	[us, ,, iraq, control, samarra, samarra, ,, ir...	[95, 1, 233, 424, 19877, 19877, 1, 233, 11, 95...
3	78606	4	Sci/Tech	Novel Approach Targets Alzheimer #39;s Develop...	A new technique may someday be able to stop Al...	novel approach targets alzheimer #39;s develop...	[novel, approach, targets, alzheimer, #, 39, ;...	[1999, 1587, 2666, 11533, 2749, 3403, 89, 1534...
4	68705	1	World	EU #39;s Prodi ready to stay on if new Brussel...	European Commission head Romano Prodi would be...	eu #39;s prodi ready to stay on if new brussel...	[eu, #, 39, ;, s, prodi, ready, to, stay, on, ...	[644, 2749, 3403, 89, 1534, 400000, 1188, 4, 1...
...
19195	105060	4	Sci/Tech	Sun buys IT services company	Sun Microsystems is buying IT	sun buys it services companv	[sun, buys, it, services,	[1662, 9911, 20, 522, 128, 4 275

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.

Ahora se define una clase MyDataset que facilita el acceso a los datos para el modelo.

```
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:

Se construye un modelo de red neuronal simple con embeddings GloVe, capas lineales y 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.

Entrenamos el modelo utilizando CrossEntropyLoss, el optimizador Adam, y se calcula la pérdida y precisión en cada época para el conjunto de entrenamiento y validación.

```

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)))

```

```

epoch 1 (train): 0%|          | 0/154 [00:00<?, ?it/s]
epoch 1 (dev): 0%|          | 0/39 [00:00<?, ?it/s]
epoch 2 (train): 0%|          | 0/154 [00:00<?, ?it/s]
epoch 2 (dev): 0%|          | 0/39 [00:00<?, ?it/s]
epoch 3 (train): 0%|          | 0/154 [00:00<?, ?it/s]
epoch 3 (dev): 0%|          | 0/39 [00:00<?, ?it/s]
epoch 4 (train): 0%|          | 0/154 [00:00<?, ?it/s]
epoch 4 (dev): 0%|          | 0/39 [00:00<?, ?it/s]
epoch 5 (train): 0%|          | 0/154 [00:00<?, ?it/s]
epoch 5 (dev): 0%|          | 0/39 [00:00<?, ?it/s]

```

Let's plot the loss and accuracy on dev:

Para tener una visualización se grafica la pérdida y precisión para cada época en los conjuntos de entrenamiento y validación.

```

import matplotlib.pyplot as plt
%matplotlib inline

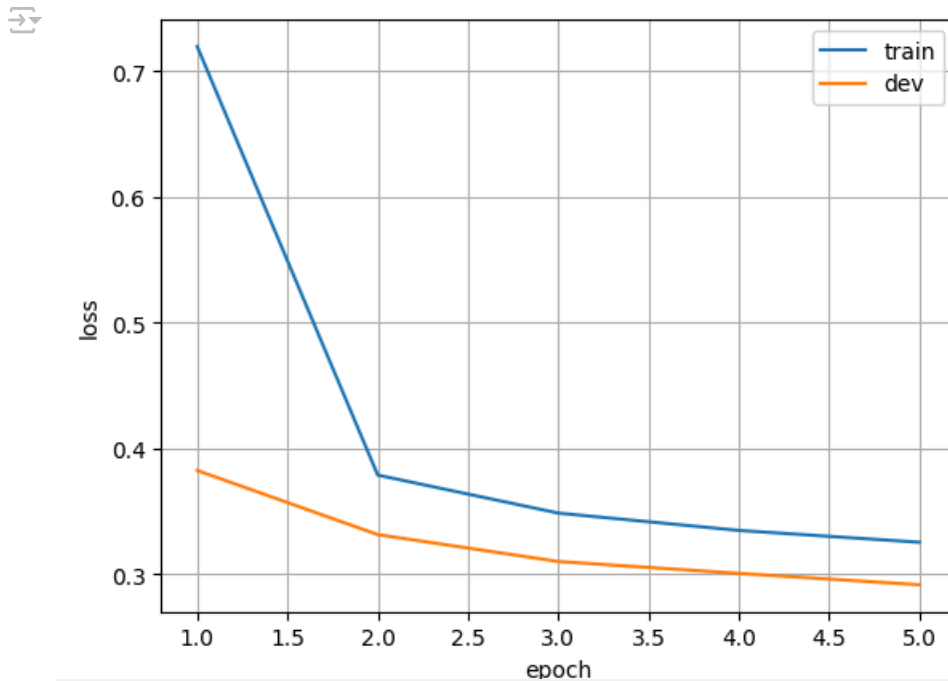
```

```
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)

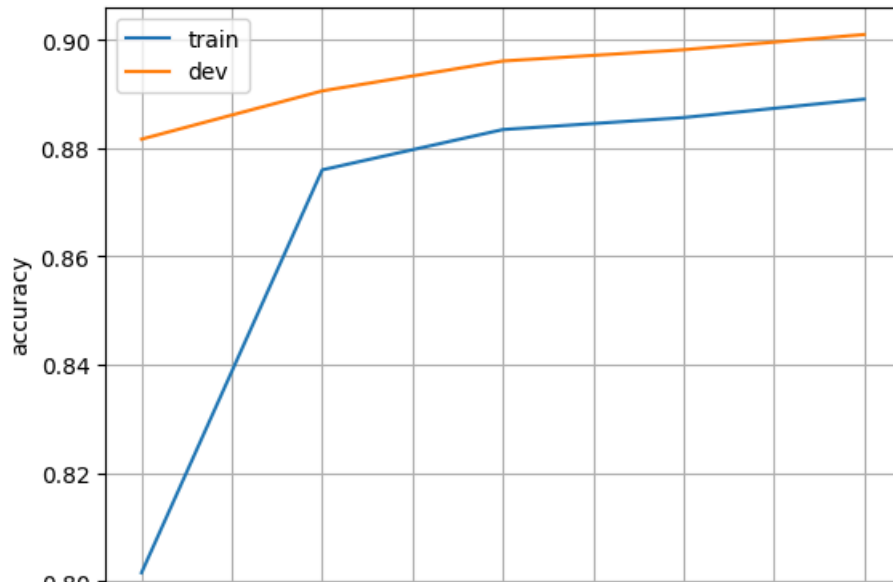
```



```

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:

epoch

Aplicamos el preprocesamiento al conjunto de prueba para tokenización y padding.

```
# repeat all preprocessing done above, this time on the test set
test_df = pd.read_csv('/kaggle/input/ag-news/ag_news_csv/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)
```



0%| | 0/7600 [00:00<?, ?it/s]

Y finalmente se usa classification_report para evaluar el rendimiento del modelo en el conjunto de prueba, generando métricas como precisión y recall para cada clase.

```
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)
```