Multiclass Text Classification with

Feed-forward Neural Networks and Word Embeddings

First, we will do some initialization.

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

# enable tqdm in pandas
tqdm.pandas()

# Permite utilizar un GPU si está disponible
use_gpu = True

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

# Pondremos que nuestra semilla de aleatorización es 1234
seed = 1234

# Establecemos nuestra semilla
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.

```
# Aquí lo que hacemos es extraer nuestros archivos CSV, donde se
encuentre nuestro conjunto de entrenamiento
train_df =
pd.read_csv('https://raw.githubusercontent.com/mhjabreel/CharCnn_Keras
/refs/heads/master/data/ag_news_csv/train.csv', header=None)
train_df.columns = ['class index', 'title', 'description']
train_df
```

```
        class index                                                title
\
0                 3  Wall St. Bears Claw Back Into the Black (Reuters)

1                 3  Carlyle Looks Toward Commercial Aerospace (Reu...

2                 3    Oil and Economy Cloud Stocks' Outlook (Reuters)

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

4                 3  Oil prices soar to all-time record, posing new...

...             ...                                                ...

119995            1  Pakistan's Musharraf Says Won't Quit as Army C...

119996            2                  Renteria signing a top-shelf deal

119997            2                    Saban not going to Dolphins yet

119998            2                                Today's NFL games

119999            2                        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 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.

```
# Extraemos las clases de nuestro conjunto
labels =
open('/kaggle/input/classes-2/ag_news_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  \
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...
```

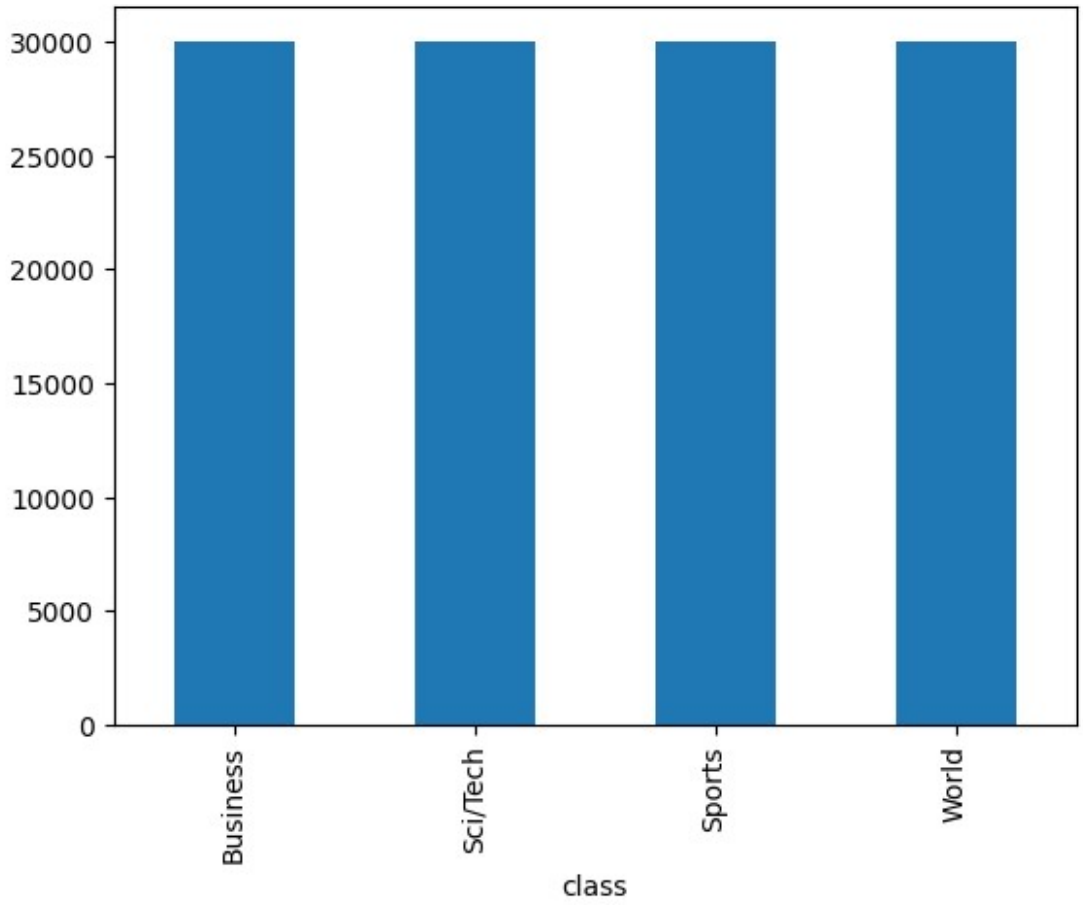
```
[120000 rows x 4 columns]
```

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

```
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".

```
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.

```
# Aquí lo que se hace es tomar las columnas de title y description, y
las convierte en minúsculas para posteriormente juntarlas.
# Finalmente, se reemplazan las dobles barras invertidas con un
espacio.
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.

```
# Aquí hacemos uso de la función word_tokenize, la cual nos permite
# dividir el texto en tokens por cada fila de la columna
from nltk.tokenize import word_tokenize

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


```
{"model_id":"5c9ae8d88a9c40b7b60f3a415ec1159c","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.

```
# Posteriormente, hacemos la carga de los vectores de palabras
preentrenados de glove  utilizando la librería gensim
# y mostramos el shape del conjunto de los vectores cargados para
tener una perspectiva de su tamaño.
from gensim.models import KeyedVectors
glove =
KeyedVectors.load_word2vec_format("/kaggle/input/classes-3/glove.6B.30
0d.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.

```
'''
Definimos la función de count_unknown_words, la cual toma listas de
tokens y vocabulary (que son las palabras conocidas, para nuestro caso
son las claves key_to_index de glove).

Para ello, se crea un contador, con el objetivo de contar los tokens
desconocidos e iterar sobre las listas de tokens en los datos.
Cada token que no está en las palabras concocidas vocabulary, es
añadido al contador.
Finalmente, devuelve el contador con los tokens desconocidos y sus
frecuencias.
'''

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

#  aquí hacemos el cálculo previamente mencionado con la func de
count_unkown_words, para los tokens del conjunto de entrenamiento.
c = count_unknown_words(train_df['tokens'], glove.key_to_index)

# Calculamos el número total de los tokens en el corpus que tenemos
total_tokens = train_df['tokens'].map(len).sum()

unk_tokens = sum(c.values())           # Número total de tokens
desconocidos
percent_unk = unk_tokens / total_tokens # Porcentaje de tokens
desconocidos del total
distinct_tokens = len(list(c))         # Número de tokens desconocidos
```

```
únicos

# imprimimos los resultados
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 10 unknown words:') # muestra las 10 palabras desconocidas
más frecuentes
for token, n in c.most_common(10):
    print(f'\t{n}\t{token}')
```


```
{"model_id":"49f5d2bd36cd4a98b82299483bfc7742","version_major":2,"vers
ion_minor":0}
```


```
total number of tokens: 5,273,465
number of unknown tokens: 66,035
number of distinct unknown tokens: 24,799
percentage of unkown tokens: 1.25%
top 10 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.

```
'''
Posteriormente, añadimos dos tokens especiales que tenemos al
vocabulario de GloVe y obtenemos sus identificadores:
[UNK] de unknown para palabras desconocidas y [PAD] de padding para el
relleno
'''

# definimos como variables los tokens especiales que queremos añadir
```

```
unk_tok = '[UNK]'
pad_tok = '[PAD]'

# inicializmamos sus valores como el promedio para unknown y un
relleno de 300 ceros para padding.
unk_emb = glove.vectors.mean(axis=0)
pad_emb = np.zeros(300)

# los añadimos a glove
glove.add_vectors([unk_tok, pad_tok], [unk_emb, pad_emb])

# obtenemos los ids de unknown y padding
unk_id = glove.key_to_index[unk_tok]
pad_id = glove.key_to_index[pad_tok]

unk_id, pad_id
```

```
(400000, 400001)
```

```
# Realizamos el split del conjunto de datos para entrenamiento y
validacion con un una proporción de 80/20.
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.

```
'''
Luego, realizamos un conjunto de palabras frecuentes en train_df,
donde solo mantendremos aquellas palabras que aparecen más
de 10 veces.
'''
threshold = 10
tokens = train_df['tokens'].explode().value_counts()
vocabulary = set(tokens[tokens > threshold].index.tolist())
print(f'vocabulary size: {len(vocabulary):,}')
```

```
vocabulary size: 17,442
```

```
# Encontramos la longitud de la lista de tokens de mayor tamaño
max_tokens = train_df['tokens'].map(len).max()

# Definimos una función get_id que regrese el unk_id para tokens no
frecuentes.
def get_id(tok):
    if tok in vocabulary:
        return glove.key_to_index.get(tok, unk_id)
```

```
    else:
        return unk_id

# Creamos una función token_ids que obtiene una lista de tokens y
regresa la lista de los token ids, realizando el relleno adecuado.
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

# añadimos nueva columna al dataframe de los token ids
train_df['token ids'] = train_df['tokens'].progress_map(token_ids)
train_df
```

{"model_id":"ab2b1ac6c9e3426c85fa8dea88177a36","version_major":2,"vers
ion_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]
```

```
# calculamos el número máximo de tokens en dev_df donde posteriormente
utlizamos token_ids para convertir los tokens a IDs numéricos.
max_tokens = dev_df['tokens'].map(len).max()
```

```
dev_df['token ids'] = dev_df['tokens'].progress_map(token_ids)
dev_df
```

{"model_id":"edda7335728144668107fd3f3694d1ce","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.

```python
# Ahora crearemos una clase denominada MyDataset con el objetivo de
trabajar con Datasets en formato de tensores y poder utilizarlos
# en PyTorch.
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:

```
'''
Posteriormente, se crea nuestro modelo de red neuronal (una FFN
básica)
'''
from torch import nn
import torch.nn.functional as F  # Importa las funciones de PyTorch,
como activaciones, desde el módulo funcional

class Model(nn.Module):  # Define una clase llamada Model que hereda
de nn.Module, la base para todos los modelos en PyTorch
    def __init__(self, vectors, pad_id, hidden_dim, output_dim,
dropout):
        super().__init__()  # Llama al constructor de la clase base
(nn.Module)

        # Verifica si los vectores están en formato tensor; si no, los
convierte
        if not torch.is_tensor(vectors):
            vectors = torch.tensor(vectors)

        # Almacena el índice de padding para usarlo en el modelo
        self.padding_idx = pad_id

        # Define la capa de embeddings usando los vectores
preentrenados y especifica el índice de padding
        self.embs = nn.Embedding.from_pretrained(vectors,
padding_idx=pad_id)

        # Define una secuencia de capas completamente conectadas
(feedforward)
        self.layers = nn.Sequential(
            nn.Dropout(dropout),  # Añade una capa de dropout para
regularización con la tasa especificada
            nn.Linear(vectors.shape[1], hidden_dim),  # Capa lineal
que mapea la dimensión de los embeddings a la dimensión oculta
            nn.ReLU(),  # Función de activación ReLU para añadir no
linealidad
            nn.Dropout(dropout),  # Otra capa de dropout
```

```
            nn.Linear(hidden_dim, output_dim),  # Capa lineal que
mapea la capa oculta a la dimensión de salida (número de clases)
        )

    def forward(self, x):
        # Crea un arreglo booleano donde los elementos de padding son
False
        not_padding = torch.isin(x, self.padding_idx, invert=True)

        # Calcula la longitud de cada ejemplo (excluyendo padding)
contando los elementos True
        lengths = torch.count_nonzero(not_padding, axis=1)

        # Obtiene los embeddings de las entradas usando la capa de
embeddings
        x = self.embs(x)

        # Suma los embeddings de cada ejemplo y los divide por su
longitud para obtener la media
        x = x.sum(dim=1) / lengths.unsqueeze(dim=1)

        # Pasa el resultado a través de las capas de la red
        output = self.layers(x)

        # Devuelve la salida del modelo (las predicciones)
        return output
```

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

```
"""
Ahora, se configura y entrena el modelo de clasificación en PyTorch.
Se define hiperparámetros, se inicializa el modelo, la función de
pérdida, el optimizador, y los loaders de datos.
Durante el entrenamiento, se calculan las pérdidas y las precisiones
tanto en el conjunto de entrenamiento como en el de validación
(desarrollo).
"""

from torch import optim  # Importa el módulo optimizador de PyTorch
para entrenar el modelo
from torch.utils.data import DataLoader  # Importa DataLoader para
cargar datos en lotes
from sklearn.metrics import accuracy_score  # Importa accuracy_score
para calcular la precisión del modelo

# Hiperparámetros
lr = 1e-3  # Tasa de aprendizaje para el optimizador
weight_decay = 0  # Decaimiento de peso para la regularización en el
```

```
optimizador
batch_size = 500  # Tamaño del lote para el entrenamiento y validación
shuffle = True  # Mezclar los datos en cada época para evitar
sobreajuste
n_epochs = 5  # Número de épocas de entrenamiento
hidden_dim = 50  # Dimensión de la capa oculta en el modelo
output_dim = len(labels)  # Dimensión de salida, igual al número de
clases
dropout = 0.1  # Tasa de dropout para regularización
vectors = glove.vectors  # Vectores de palabras preentrenados (GloVe)

# Inicialización del modelo, la función de pérdida, el optimizador, y
los loaders de datos
model = Model(vectors, pad_id, hidden_dim, output_dim,
dropout).to(device)  # Crea el modelo y lo envía al dispositivo (CPU o
GPU)
loss_func = nn.CrossEntropyLoss()  # Define la función de pérdida de
entropía cruzada para clasificación
optimizer = optim.Adam(model.parameters(), lr=lr,
weight_decay=weight_decay)  # Usa el optimizador Adam con la tasa de
aprendizaje y el decaimiento de peso

# Crea el conjunto de entrenamiento y su DataLoader
train_ds = MyDataset(train_df['token ids'], train_df['class index'] -
1)  # Prepara los datos de entrenamiento con sus etiquetas
train_dl = DataLoader(train_ds, batch_size=batch_size,
shuffle=shuffle)  # Define el DataLoader para cargar datos en lotes de
entrenamiento

# Crea el conjunto de validación y su DataLoader
dev_ds = MyDataset(dev_df['token ids'], dev_df['class index'] - 1)  #
Prepara los datos de validación con sus etiquetas
dev_dl = DataLoader(dev_ds, batch_size=batch_size, shuffle=shuffle)  #
Define el DataLoader para cargar datos en lotes de validación

# Inicializa listas para almacenar la pérdida y precisión en
entrenamiento y validación
train_loss = []  # Pérdidas en el conjunto de entrenamiento
train_acc = []  # Precisión en el conjunto de entrenamiento
dev_loss = []  # Pérdidas en el conjunto de validación
dev_acc = []  # Precisión en el conjunto de validación

# Entrenamiento del modelo
for epoch in range(n_epochs):  # Bucle sobre cada época
    losses = []  # Almacena pérdidas por lote en la época actual
    gold = []  # Almacena etiquetas reales
    pred = []  # Almacena predicciones del modelo
    model.train()  # Pone el modelo en modo de entrenamiento

    for X, y_true in tqdm(train_dl, desc=f'epoch {epoch+1} (train)'):
```

```
# Bucle sobre cada lote de entrenamiento
        model.zero_grad()  # Limpia los gradientes de la iteración
anterior

        X = X.to(device)  # Envía los datos de entrada al dispositivo
        y_true = y_true.to(device)  # Envía las etiquetas al
dispositivo

        y_pred = model(X)  # Calcula las predicciones del modelo
        loss = loss_func(y_pred, y_true)  # Calcula la pérdida
comparando las predicciones con las etiquetas

        losses.append(loss.detach().cpu().item())  # Almacena la
pérdida del lote actual
        gold.append(y_true.detach().cpu().numpy())  # Almacena las
etiquetas reales
        pred.append(np.argmax(y_pred.detach().cpu().numpy(), axis=1))
# Almacena las predicciones del modelo

        loss.backward()  # Calcula los gradientes mediante
retropropagación
        optimizer.step()  # Actualiza los parámetros del modelo usando
el optimizador

    # Almacena la pérdida y precisión promedio en la época actual
    train_loss.append(np.mean(losses))  # Pérdida promedio en
entrenamiento
    train_acc.append(accuracy_score(np.concatenate(gold),
np.concatenate(pred)))  # Precisión en entrenamiento

    model.eval()  # Cambia el modelo a modo de evaluación
    with torch.no_grad():  # Desactiva el cálculo de gradientes
        losses = []  # Almacena pérdidas por lote en validación
        gold = []  # Almacena etiquetas reales en validación
        pred = []  # Almacena predicciones del modelo en validación

        for X, y_true in tqdm(dev_dl, desc=f'epoch {epoch+1} (dev)'):
# Bucle sobre cada lote de validación
            X = X.to(device)  # Envía los datos de entrada al
dispositivo
            y_true = y_true.to(device)  # Envía las etiquetas al
dispositivo

            y_pred = model(X)  # Calcula las predicciones del modelo
            loss = loss_func(y_pred, y_true)  # Calcula la pérdida en
validación

            losses.append(loss.cpu().item())  # Almacena la pérdida
del lote actual en validación
```

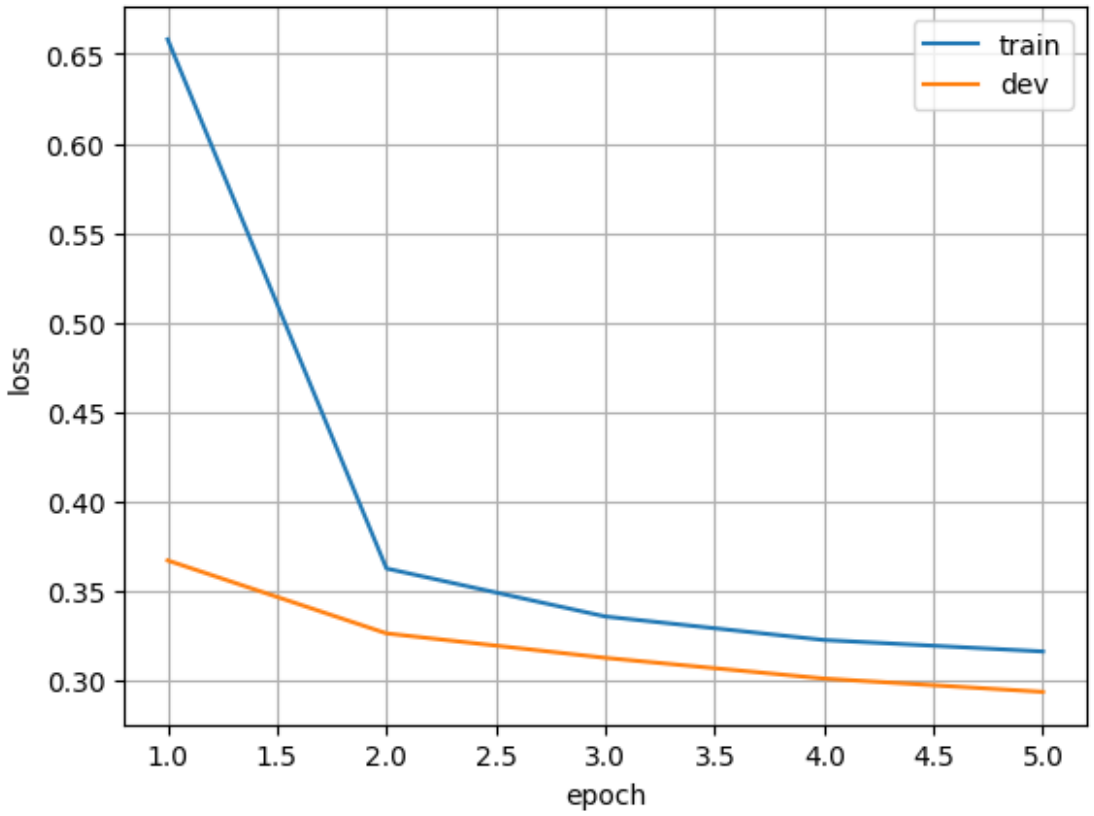
```
            gold.append(y_true.cpu().numpy())  # Almacena las
etiquetas reales en validación
            pred.append(np.argmax(y_pred.cpu().numpy(), axis=1))  #
Almacena las predicciones del modelo en validación

        # Almacena la pérdida y exactitud promedio en validación
        dev_loss.append(np.mean(losses))  # Pérdida promedio en
validación
        dev_acc.append(accuracy_score(np.concatenate(gold),
np.concatenate(pred)))  # Exactitud en validación
```


```
{"model_id":"0b98c5f8489343068ac9d80167104de0","version_major":2,"vers
ion_minor":0}

{"model_id":"5676a4bc589d40549693f8d6351e03d9","version_major":2,"vers
ion_minor":0}

{"model_id":"5ebf72ffbe23475e88c6062ec5d835d8","version_major":2,"vers
ion_minor":0}

{"model_id":"f25214bbe4e24d058d32fd3077542c8c","version_major":2,"vers
ion_minor":0}

{"model_id":"7488f556c7b64293816e597ac98eb073","version_major":2,"vers
ion_minor":0}

{"model_id":"0055a7c92a874fd9b1e99dacb1bfc74b","version_major":2,"vers
ion_minor":0}

{"model_id":"57851d9a22fe4c58941c7259fee8dfec","version_major":2,"vers
ion_minor":0}

{"model_id":"135b608fcf4842a29c99c85c127596de","version_major":2,"vers
ion_minor":0}

{"model_id":"3c3b2ec6c0ce4ce686e286987389d533","version_major":2,"vers
ion_minor":0}

{"model_id":"443d8379a0eb498096e80999a95086f9","version_major":2,"vers
ion_minor":0}
```

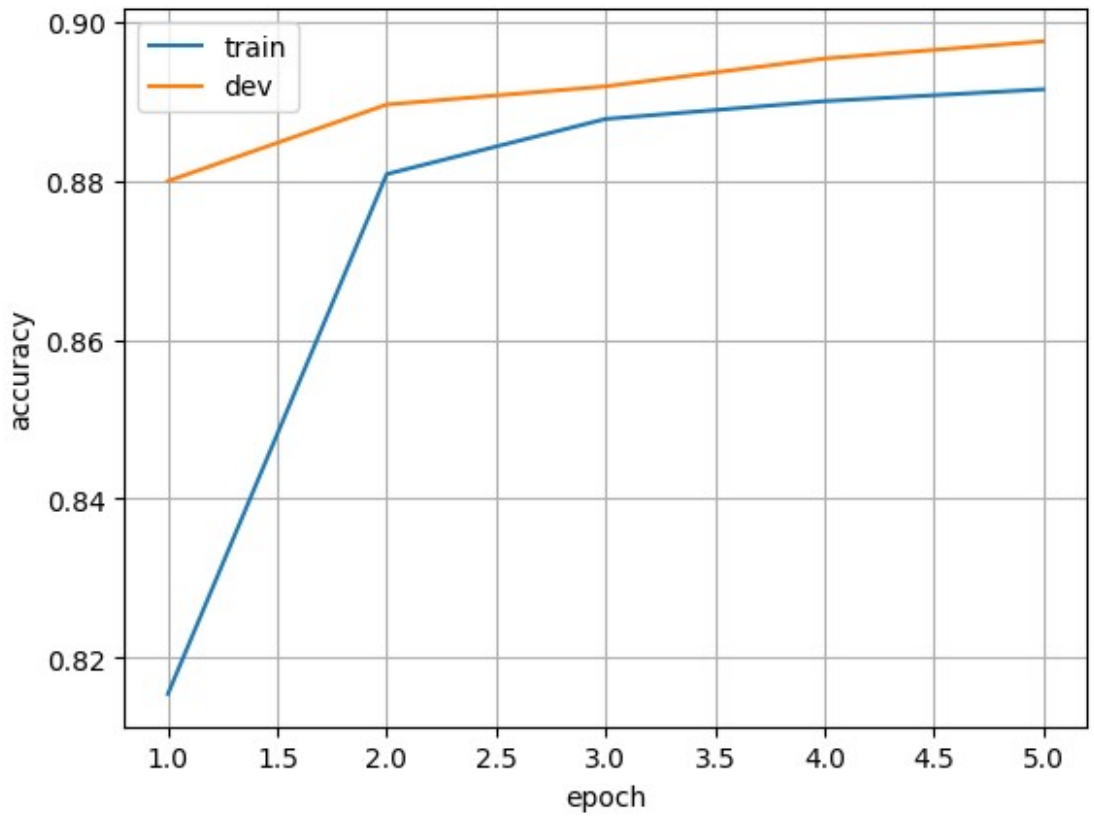

Let's plot the loss and accuracy on dev:

```
# Ahora,  crearemos la gráfica de loss para visualizar el rendimiento
del modelo en el conjunto de entrenamiento y validación en las épocas

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)
```




```
# Luego, graficamos las exactitudes en entrenamiento y validación del modelo
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:

```
# Se repite todo lo realizado de prepocesamiento, pero en el conjunto
de prueba
test_df =
pd.read_csv('https://raw.githubusercontent.com/mhjabreel/CharCnn_Keras
/refs/heads/master/data/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)

{"model_id":"512d5bad2b7447e68e9bd3c26d34a133","version_major":2,"vers
ion_minor":0}

{"model_id":"91733288acf84756b513b12db8b64d89","version_major":2,"vers
ion_minor":0}

from sklearn.metrics import classification_report

# evaluamos nuestro modelo
model.eval()
```

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

# sin guardar los gradientes, imprimimos los resultados relevantes
(precision, recall, f1-score, support)
with torch.no_grad():
    for X, _ in tqdm(data_loader):
        X = X.to(device)
        # se predice una clase por ejemplo
        y = torch.argmax(model(X), dim=1)
        # convertimos de tensor a np array
        y_pred.append(y.cpu().numpy())
        # imprimimos resultados
    print(classification_report(dataset.y, np.concatenate(y_pred),
target_names=labels))
```

{"model_id":"5379d4961b704af18b7203817fd4ea09","version_major":2,"vers
ion_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      0.86      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
```

Como podemos observar, los resultados son buenos y adecuados, pues un f1-score del 90% marca un modelo adecuadamente# Pipeline y Proceso Realizado

Este pipeline implementa un modelo de clasificación de texto multiclase utilizando redes neuronales feed-forward y embeddings de palabras (GloVe) en PyTorch. A continuación se detallan las etapas del proceso:

1. **Configuración Inicial**:
   - Importación de librerías clave como `torch`, `pandas`, `nltk` y `gensim`.
   - Configuración del dispositivo (GPU o CPU).
   - Establecimiento de una semilla aleatoria para asegurar reproducibilidad.
2. **Carga y Preprocesamiento de Datos**:
   - Carga del conjunto de datos de AG's News.
   - Conversión de títulos y descripciones a minúsculas y eliminación de caracteres especiales.
   - Tokenización de texto utilizando `word_tokenize()` de `nltk`.

3. **Construcción del Vocabulario y Embeddings**:
   - Se cargan embeddings de palabras preentrenados de GloVe.
   - Cálculo de la frecuencia de tokens en el corpus y creación de un vocabulario filtrado.
   - Añaden dos embeddings especiales: `[UNK]` para palabras desconocidas y `[PAD]` para rellenar secuencias a una longitud uniforme.
4. **Codificación y Relleno de Secuencias**:
   - Conversión de tokens en listas de identificadores de tokens (`token ids`) para cada texto, incluyendo padding.
   - Dividir los datos en conjunto de entrenamiento y validación.
5. **Definición del Modelo**:
   - Se implementa una red neuronal feed-forward (FFN) con una capa de embeddings, una capa oculta con activación ReLU, y una capa de salida para la predicción de clases.
   - Configuración de dropout para regularización.
6. **Entrenamiento del Modelo**:
   - Uso de entropía cruzada como función de pérdida y optimización con Adam.
   - Se realiza un entrenamiento en mini-batches con retropropagación, optimización, y cálculo de pérdida y precisión por época.
7. **Evaluación del Modelo**:
   - Se calcula la pérdida y precisión en los conjuntos de entrenamiento y validación tras cada época.
   - Al final, se utiliza el conjunto de prueba para evaluar la precisión, exhaustividad y puntuación F1 del modelo.

Este pipeline permite implementar y evaluar una red feed-forward con embeddings para clasificación de texto, proporcionando resultados efectivos en la tarea de clasificación multiclase. rígido.