Implementación de CAMELON

CAMELON: A System for Crime Metadata Extraction and Spatiotemporal Visualization From Online News Articles

https://ieeexplore.ieee.org/document/10424974

Esta implementación solo cubre la parte de crime-classification

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
!unzip -q /content/drive/MyDrive/CIENCIA\ DE\
DATOS/tweets_predict10k_2020.zip
```

```
import pandas as pd
ruta = "/content/tweets_predict10k_2020.csv"
df = pd.read_csv(ruta)
```

```
<ipython-input-1-976793c0ea66>:3: DtypeWarning: Columns (43) have
mixed types. Specify dtype option on import or set low_memory=False.
  df = pd.read_csv(ruta)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101630 entries, 0 to 101629
Data columns (total 44 columns):
 #   Column                         Non-Null Count   Dtype
---  ------                         --------------   -----
 0   Unnamed: 0.1                   101630 non-null  int64
 1   Unnamed: 0                     101630 non-null  int64
 2   ID                             101630 non-null  int64
 3   url                            101630 non-null  object
 4   Date                           101630 non-null  object
 5   lang                           101630 non-null  object
 6   Description                    101630 non-null  object
 7   replyCount                     101630 non-null  int64
 8   retweetCount                   101630 non-null  int64
 9   likeCount                      101630 non-null  int64
 10  quoteCount                     101630 non-null  int64
 11  bookmarkedCount                101630 non-null  int64
 12  conversationId                 101630 non-null  int64
 13  hashtags                       101630 non-null  object
 14  cashtags                       101630 non-null  object
 15  mentionedUsers                 101630 non-null  object
 16  links                          101630 non-null  object
 17  source                      101630 non-null  object
 18  sourceUrl                   101630 non-null  object
 19  sourceLabel                 101630 non-null  object
 20  possibly_sensitive          46600 non-null   object
 21  _type                       101630 non-null  object
 22  user_id                     101630 non-null  int64
 23  user_username               101630 non-null  object
 24  user_followersCount         101630 non-null  int64
 25  inReplyToUser_username      30204 non-null   object
 26  inReplyToUser_displayname   30199 non-null   object
 27  inReplyToUser__type         30205 non-null   object
 28  Year                        101630 non-null  int64
 29  Month                       101630 non-null  int64
 30  dayOfWeek                   101630 non-null  int64
 31  dayOfMonth                  101630 non-null  int64
 32  dayOfYear                   101630 non-null  int64
 33  weekOfMonth                 101630 non-null  int64
 34  weekOfYear                  101630 non-null  int64
 35  Hour                        101630 non-null  int64
 36  Minute                      101630 non-null  int64
 37  Hour_Zone                   101630 non-null  object
 38  BusinessHour                101630 non-null  int64
 39  Weekend                     101630 non-null  int64
 40  Season                      101630 non-null  object
 41  Holiday                     101630 non-null  int64
 42  CleanDescription            101578 non-null  object
 43  Category                    15000 non-null   object
dtypes: int64(23), object(21)
memory usage: 34.1+ MB
```

```
df["Category"].unique()
```

```
array(['SOCIAL_COMMENTARY', 'NON_CRIME_RELATED', 'KIDNAPPING',
'HOMICIDE',
       'ASSAULT', 'NEWS_MEDIA_MENTION', 'THEFT', 'BURGLARY',
'BATTERY',
       'WEAPONS VIOLATION', 'OTHER OFFENSE', 'OFFENSE INVOLVING
CHILDREN',
       'PUBLIC PEACE VIOLATION', 'INTIMIDATION', 'ARSON', 'AMBIGUOUS',
       'CRIMINAL DAMAGE', 'CRIMINAL SEXUAL ASSAULT', 'ROBBERY',
       'NARCOTICS', 'INTERFERENCE WITH PUBLIC OFFICER',
       'FICTIONAL_CONTENT', 'MOTOR VEHICLE THEFT', 'DECEPTIVE
PRACTICE',
       'CRIMINAL TRESPASS', 'SEX OFFENSE', 'LIQUOR LAW VIOLATION',
       'CONCEALED CARRY LICENSE VIOLATION', 'THREAT',
       'CRIM SEXUAL ASSAULT', 'GAMBLING', 'HUMAN TRAFFICKING',
       'PROSTITUTION', 'HUMOR_OR_SATIRE', nan], dtype=object)
```

```
# Lista de clases irrelevantes o no relacionadas a crimen
non_crime_classes = [
    "FICTIONAL_CONTENT",
    "SOCIAL_COMMENTARY",
    "NEWS_MEDIA_MENTION",
    "HUMOR_OR_SATIRE",
    "NON_CRIME_RELATED",
    "AMBIGUOUS"
]

# Reemplazar esas clases por una sola etiqueta: "NON_CRIME"
df["Category"] = df["Category"].replace(non_crime_classes,
"NON_CRIME")

# Mapeo de categorías redundantes a categorías simplificadas
category_mapping = {
    "CRIMINAL SEXUAL ASSAULT": "SEXUAL ASSAULT",
    "CRIM SEXUAL ASSAULT": "SEXUAL ASSAULT",
    "SEX OFFENSE": "SEXUAL ASSAULT",

    "NARCOTICS": "DRUG OFFENSE",
    "OTHER NARCOTIC VIOLATION": "DRUG OFFENSE",

    "BATTERY": "ASSAULT",
    "ASSAULT": "ASSAULT",

    "BURGLARY": "THEFT",
    "ROBBERY": "THEFT",
    "THEFT": "THEFT",
    "MOTOR VEHICLE THEFT": "THEFT",

    "LIQUOR LAW VIOLATION": "WEAPONS/LIQUOR VIOLATION",
    "CONCEALED CARRY LICENSE VIOLATION": "WEAPONS/LIQUOR VIOLATION",
    "WEAPONS VIOLATION": "WEAPONS/LIQUOR VIOLATION",

    "PUBLIC PEACE VIOLATION": "PUBLIC ORDER OFFENSE",
    "INTERFERENCE WITH PUBLIC OFFICER": "PUBLIC ORDER OFFENSE",
    "INTIMIDATION": "PUBLIC ORDER OFFENSE",

    "OBSCENITY": "INDECENCY",
    "PUBLIC INDECENCY": "INDECENCY"
}

# Aplicar el mapeo
df["Category"] = df["Category"].replace(category_mapping)

df["Category"].unique()
```

```
array(['NON_CRIME', 'KIDNAPPING', 'HOMICIDE', 'ASSAULT', 'THEFT',
       'WEAPONS/LIQUOR VIOLATION', 'OTHER OFFENSE',
       'OFFENSE INVOLVING CHILDREN', 'PUBLIC ORDER OFFENSE', 'ARSON',
       'CRIMINAL DAMAGE', 'SEXUAL ASSAULT', 'DRUG OFFENSE',
       'DECEPTIVE PRACTICE', 'CRIMINAL TRESPASS', 'THREAT',
'GAMBLING',
       'HUMAN TRAFFICKING', 'PROSTITUTION', nan], dtype=object)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101630 entries, 0 to 101629
Data columns (total 44 columns):
 #   Column                     Non-Null Count   Dtype
---  ------                     --------------   -----
 0   Unnamed: 0.1               101630 non-null  int64
 1   Unnamed: 0                 101630 non-null  int64
 2   ID                         101630 non-null  int64
 3   url                        101630 non-null  object
 4   Date                       101630 non-null  object
 5   lang                       101630 non-null  object
 6   Description                101630 non-null  object
 7   replyCount                 101630 non-null  int64
 8   retweetCount               101630 non-null  int64
 9   likeCount                  101630 non-null  int64
 10  quoteCount                 101630 non-null  int64
 11  bookmarkedCount            101630 non-null  int64
 12  conversationId             101630 non-null  int64
 13  hashtags                   101630 non-null  object
 14  cashtags                   101630 non-null  object
 15  mentionedUsers             101630 non-null  object
 16  links                      101630 non-null  object
 17  source                     101630 non-null  object
 18  sourceUrl                  101630 non-null  object
 19  sourceLabel                101630 non-null  object
 20  possibly_sensitive         46600 non-null   object
 21  _type                      101630 non-null  object
 22  user_id                    101630 non-null  int64
 23  user_username              101630 non-null  object
 24  user_followersCount        101630 non-null  int64
 25  inReplyToUser_username     30204 non-null   object
 26  inReplyToUser_displayname  30199 non-null   object
 27  inReplyToUser__type        30205 non-null   object
 28  Year                       101630 non-null  int64
 29  Month                      101630 non-null  int64
 30  dayOfWeek                  101630 non-null  int64
 31  dayOfMonth                 101630 non-null  int64
 32  dayOfYear                  101630 non-null  int64
 33  weekOfMonth                101630 non-null  int64
 34  weekOfYear                 101630 non-null  int64
 35  Hour                       101630 non-null  int64
 36  Minute                     101630 non-null  int64
 37  Hour_Zone                  101630 non-null  object
 38  BusinessHour               101630 non-null  int64
 39  Weekend                    101630 non-null  int64
 40  Season                     101630 non-null  object
 41  Holiday                    101630 non-null  int64
 42  CleanDescription           101578 non-null  object
 43  Category                   15000 non-null   object
dtypes: int64(23), object(21)
memory usage: 34.1+ MB
```

```python
df["Category"].value_counts()
```

```
Category
NON_CRIME                     5417
HOMICIDE                      4658
ASSAULT                       2291
PUBLIC ORDER OFFENSE           782
WEAPONS/LIQUOR VIOLATION       560
OTHER OFFENSE                  359
THEFT                          232
OFFENSE INVOLVING CHILDREN     184
DRUG OFFENSE                   166
KIDNAPPING                     118
ARSON                          110
CRIMINAL DAMAGE                 39
SEXUAL ASSAULT                  39
DECEPTIVE PRACTICE              18
HUMAN TRAFFICKING               13
PROSTITUTION                     6
GAMBLING                         4
CRIMINAL TRESPASS                2
THREAT                           2
Name: count, dtype: int64
```

```python
import pandas as pd
import torch
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from datasets import Dataset
from transformers import (
    XLMRobertaTokenizerFast,
    XLMRobertaForSequenceClassification,
    Trainer,
    TrainingArguments,
)
from torch.nn import CrossEntropyLoss

# Filtrar filas con categoría válida
df_filtered = df[df['Category'].notna()].copy()
df_filtered = df_filtered.dropna(subset=['CleanDescription'])

# Codificar las categorías a etiquetas numéricas
le = LabelEncoder()
```

```python
df_filtered['label'] = le.fit_transform(df_filtered['Category'])
num_labels = len(le.classes_)

class_counts = df_filtered['label'].value_counts().sort_index()
weights = 1.0 / class_counts
weights = weights / weights.sum()  # normalizar pesos
weights = torch.tensor(weights.values, dtype=torch.float).to('cuda' if
torch.cuda.is_available() else 'cpu')

# --- Filtrar clases con pocas muestras ---
min_samples = 100
class_counts = df_filtered['label'].value_counts()
valid_labels = class_counts[class_counts >= min_samples].index
df_filtered = df_filtered[df_filtered['label'].isin(valid_labels)]

# --- Reindexar etiquetas ---
le = LabelEncoder()
df_filtered['label'] = le.fit_transform(df_filtered['label'])

# --- División en entrenamiento y prueba ---
train_df, test_df = train_test_split(
    df_filtered[['CleanDescription', 'label']],
    test_size=0.2,
    stratify=df_filtered['label'],
    random_state=42
)
```

```
<ipython-input-12-14f84e9daea5>:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df_filtered['label'] = le.fit_transform(df_filtered['label'])
```

```python
tokenizer = XLMRobertaTokenizerFast.from_pretrained("xlm-roberta-
base")

# --- Tokenización segura ---
def tokenize_function(examples):
    return tokenizer(
        examples["CleanDescription"],
        truncation=True,
        padding='max_length',
        max_length=128
    )
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/
_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
```

```
To authenticate with the Hugging Face Hub, create a token in your
settings tab (https://huggingface.co/settings/tokens), set it as
secret in your Google Colab and restart your session.
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to
access public models or datasets.
  warnings.warn(
```

```
# --- Eliminar índice heredado (muy importante) ---
train_df = train_df.reset_index(drop=True)
test_df = test_df.reset_index(drop=True)

# --- Conversión a HuggingFace Datasets ---
train_dataset = Dataset.from_pandas(train_df)
test_dataset = Dataset.from_pandas(test_df)

tokenized_train = train_dataset.map(tokenize_function, batched=True)
tokenized_test = test_dataset.map(tokenize_function, batched=True)
```

{"model_id":"651755be92dc4c7d8473144548c45151","version_major":2,"version_minor":0}

{"model_id":"da07a4031e3c4e3991567d3d5dc91cb8","version_major":2,"version_minor":0}

```
model = XLMRobertaForSequenceClassification.from_pretrained("xlm-roberta-base", num_labels=num_labels)
```

```
Some weights of XLMRobertaForSequenceClassification were not
initialized from the model checkpoint at xlm-roberta-base and are
newly initialized: ['classifier.dense.bias',
'classifier.dense.weight', 'classifier.out_proj.bias',
'classifier.out_proj.weight']
You should probably TRAIN this model on a down-stream task to be able
to use it for predictions and inference.
```

```
class WeightedLossTrainer(Trainer):
    def __init__(self, class_weights=None, *args, **kwargs):
        self.class_weights = class_weights
        super().__init__(*args, **kwargs)

    def compute_loss(self, model, inputs, return_outputs=False,
**kwargs):  # <- ¡Aquí el cambio!
        labels = inputs.get("labels")
        outputs = model(**inputs)
        logits = outputs.logits

        loss_fct = CrossEntropyLoss(weight=self.class_weights)
        loss = loss_fct(logits.view(-1, model.config.num_labels),
labels.view(-1))
```

```
        return (loss, outputs) if return_outputs else loss

from sklearn.metrics import accuracy_score, f1_score, precision_score,
recall_score

def compute_metrics(eval_pred):
    logits, labels = eval_pred
    predictions = logits.argmax(axis=-1)

    return {
        "accuracy": accuracy_score(labels, predictions),
        "f1": f1_score(labels, predictions, average="weighted"),
        "precision": precision_score(labels, predictions,
average="weighted"),
        "recall": recall_score(labels, predictions,
average="weighted"),
    }

training_args = TrainingArguments(
    output_dir="./results",
    eval_strategy="epoch",
    per_device_train_batch_size=64,
    per_device_eval_batch_size=64,
    num_train_epochs=20,
    save_strategy="epoch",
    logging_dir="./logs",
    logging_steps=10,
    report_to="none",
    load_best_model_at_end=True,
)

trainer = WeightedLossTrainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_train,
    eval_dataset=tokenized_test,
    tokenizer=tokenizer,
    class_weights=weights,
    compute_metrics=compute_metrics
)

trainer.train()
```

```
<ipython-input-16-76e60c31d594>:4: FutureWarning: `tokenizer` is
deprecated and will be removed in version 5.0.0 for
`WeightedLossTrainer.__init__`. Use `processing_class` instead.
  super().__init__(*args, **kwargs)
```

```
<IPython.core.display.HTML object>
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/
_classification.py:1565: UndefinedMetricWarning: Precision is ill-
defined and being set to 0.0 in labels with no predicted samples. Use
`zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
```

```
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
```

```
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
Could not locate the best model at
./results/checkpoint-930/pytorch_model.bin, if you are running a
distributed training on multiple nodes, you should activate `--
save_on_each_node`.
```

```
TrainOutput(global_step=3720, training_loss=0.10442525179636093,
metrics={'train_runtime': 6442.183, 'train_samples_per_second':
36.935, 'train_steps_per_second': 0.577, 'total_flos':
1.565355006091776e+16, 'train_loss': 0.10442525179636093, 'epoch':
20.0})
```

```
model_path = "/content/drive/MyDrive/CIENCIA DE DATOS/modelos-xmlr"

# Guardar el modelo y el tokenizer en la ruta especificada
trainer.save_model(model_path)
tokenizer.save_pretrained(model_path)
```

```
('/content/drive/MyDrive/CIENCIA DE
DATOS/modelos-xmlr/tokenizer_config.json',
 '/content/drive/MyDrive/CIENCIA DE
DATOS/modelos-xmlr/special_tokens_map.json',
 '/content/drive/MyDrive/CIENCIA DE
DATOS/modelos-xmlr/sentencepiece.bpe.model',
 '/content/drive/MyDrive/CIENCIA DE
DATOS/modelos-xmlr/added_tokens.json',
 '/content/drive/MyDrive/CIENCIA DE
DATOS/modelos-xmlr/tokenizer.json')
```

```
trainer.evaluate()
```

```
<IPython.core.display.HTML object>
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/metrics/
_classification.py:1565: UndefinedMetricWarning: Precision is ill-
defined and being set to 0.0 in labels with no predicted samples. Use
`zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
```

```
{'eval_loss': 0.6859170794487,
 'eval_accuracy': 0.6292436974789916,
 'eval_f1': 0.5548542189524683,
 'eval_precision': 0.5041606520314937,
 'eval_recall': 0.6292436974789916,
 'eval_runtime': 17.0244,
 'eval_samples_per_second': 174.749,
 'eval_steps_per_second': 2.761,
 'epoch': 20.0}
```