UM6P at SemEval-2023 Task 3: News genre classification based on transformers, graph convolution networks and number of sentences

Hamza Alami[1], **Abdessamad Benlahbib**[2], **Abdelkader El Mahdaouy**[3], **Ismail Berrada**[1]
[1]School of Computer Science, Mohammed VI Polytechnic University, Morocco
[2]Laboratory of Informatics, Signals, Automatics, and Cognitivism (LISAC),
Faculty of Sciences Dhar El Mahraz, Sidi Mohammed Ben Abdellah University, Morocco
[3]Modeling, Simulation and Data Analysis, Mohammed VI Polytechnic University, Morocco
{firstname.lastname}@{um6p.ma[1,3], usmba.ac.ma[2]}

Abstract

This paper presents our proposed approach for English document genre classification in the context of SemEval-2023 Task 3, Subtask 1. Our method uses an ensemble technique to combine four distinct model predictions: Longformer, RoBERTa, GCN, and a sentence number-based model. Each model is optimized on simple and easy-to-understand objectives. We provide snippets of code that define each model to make the reading experience better. Our method ranked 12th in document genre (subtask 1) classification for English texts.

1 Introduction

Detecting the genre, framing, and persuasion techniques in online news articles is a crucial task in media analysis, with applications ranging from content curation to political campaign analysis. With the increasing diversity of online news sources and the proliferation of multiple languages, the ability to perform such analysis in a multilingual setup is becoming ever more important. In this context, the authors (Piskorski et al., 2023) proposed a new competition in the context of the SemEval 2023 shared task, which aims to encourage the use of artificial intelligence (AI) in media analysis. The competition's dataset is not publicly available and covers various aspects of what makes a text persuasive, such as the genre (opinion, report, or satire), framing (which key aspects are highlighted), and rhetoric (persuasion techniques used to influence the reader). The shared task includes three subtasks: news genre categorization, framing detection, and persuasion techniques detection. The participants can choose to take part in any number of subtask-language pairs. The dataset is available in six languages, and participants can train their systems using the data for all languages in a multilingual setup.

As part of our participation in the SemEval 2023 task 3, we will be focusing on the News Genre Categorisation (subtask 1), working specifically in a monolingual setting with English language texts. Our aim is to develop and test systems that can accurately classify news articles according to their genre. Through our participation in this shared task, we hope to contribute to the growing field of AI-based media analytics.

In this paper, we present our method for English document genre classification, which combines four different models to achieve promising results. Our method integrates 1) Longformer and RoBERTa, two deep learning models based on Transformer architecture that have demonstrated strong performance in various NLP tasks (Vaswani et al., 2017; El Mekki et al., 2022, 2020), with 2) a Graph Convolutional Network (GCN) that captures the relationships between different parts of the input text (Yao et al., 2018; Liu et al., 2020). Additionally, we incorporate a sentence number-based model based on the count of the number of sentences in each document. Finally, we use a voting mechanism to determine the most likely genre. Our method scored 39.35% and ranked 12th in the English test leaderboard.

2 Method

Our method aims to build an English document genre classifier leveraging an ensemble model based on four distinct models: Longformer, RoBERTa, GCN, and a sentence number-based model. The following paragraphs describe the proposed method.

2.1 Longformer model

Longformer (Beltagy et al., 2020) is a transformer-based model (Vaswani et al., 2017) designed to process long sequences of text. This model uses an attention mechanism that allows to focus on specific parts of the input sequence, while ignoring others. The Longformer model applies a sliding window approach to process long sequences, i.e.,

Proceedings of the The 17th International Workshop on Semantic Evaluation (SemEval-2023), pages 856–861

instead of processing the entire sequence at once, the model processes the sequence in chunks or windows. However, for some tasks such as text classification, the model aggregates the representation of the whole sequence into a special token (*[CLS]* in case of BERT). In this case the sliding window approach is not flexible enough to learn efficient representations for the special token. Thus, a global attention mechanism on a few pre-selected input locations is added. For instance, the special token *[CLS]* attends to all tokens across the sequence, and all tokens in the sequence attend to it. Thus, long-range dependencies are captured from the input sequence. In addition to the global attention mechanism, the Longformer model applies a local attention mechanism to attend to nearby positions within the current window. This local attention mechanism helps the model capture short-range dependencies from the input sequence.

```
from transformers import AutoTokenizer,
    LongformerForSequenceClassification

tokenizer = AutoTokenizer.
    from_pretrained("allenai/Longformer-
    base-4096")
model =
    LongformerForSequenceClassification.
    from_pretrained("allenai/Longformer-
    base-4096", num_labels=3)
```

Listing 1: Code definition of the Longformer model

We used the Hugging Face library (Wolf et al., 2020) to use a pre-trained Longformer model that started from the RoBERTa (Liu et al., 2019) checkpoint and pre-trained for Masked Language Modeling (MLM) on long documents. It supports sequences up to 4,096 in length. Listing 1 presents the code of the used Longformer model. Since the model has the ability to process large documents, we only tokenized documents with the pre-trained tokenizer and then fed them to the Longformer model.

2.2 RoBERTa

RoBERTa (Robustly Optimized BERT approach) (Liu et al., 2019) is a transformer-based language model developed by Facebook AI Research in 2019. It is an extension of BERT (Devlin et al., 2019) model, which was released by Google in 2018.

Like BERT, RoBERTa is a pre-trained language model that learns to represent texts by training on large amounts of unlabeled data. The model is trained on an MLM task, where random tokens in the input sequence are masked and the model is trained to predict the original tokens. RoBERTa improves upon BERT by using a larger training corpus, more training data augmentation techniques, and a longer training time. Specifically, RoBERTa is trained on a much larger corpus of texts than BERT, including BooksCorpus (16GB), CC-NEWS (76GB), OPENWEBTEXT (38GB), and STORIES (31GB). RoBERTa also uses a dynamic masking approach during training, where each training example is randomly masked multiple times, which helps the model learn to generalize better.

XLM-RoBERTa (Conneau et al., 2019) is a multilingual version of RoBERTa, pre-trained on 2.5TB of filtered CommonCrawl data containing 100 languages. We also used the Hugging Face library for XLM-RoBERTa. Listing 2 presents the code of the used RoBERTa model. However, the model can accept only 512 tokens as input. Therefore, if the tokenized document contains a large number of tokens, we keep the first 256 tokens and the last 256 tokens. We argue that the first tokens contain the title of a document if it exists and the last part of a document can contain valuable information for the final decision (according to section 4.6 in the annotation guidelines[1]).

```
from transformers import AutoTokenizer,
    XLMRobertaForSequenceClassification

tokenizer = AutoTokenizer.
    from_pretrained("xlm-roberta-base")
model =
    XLMRobertaForSequenceClassification.
    from_pretrained("xlm-roberta-base",
    num_labels=3)
```

Listing 2: Code definition of the RoBERTa model

2.3 Graph Convolutional Networks

Graph Convolutional Networks (GCNs) (Kipf and Welling, 2017) are a type of neural network architectures that can be used to perform classification on text documents. In order to apply GCNs to text classification, the text documents are first represented as a graph structure, where each node in the graph represents a word or a sequence of words (e.g., a sentence), and the edges between nodes represent relationships between them. To create the graph structure, we first remove stop words from the text. After that, we extract a set of features for

[1] https://knowledge4policy.ec.europa.eu/sites/default/files/JRC132862_technical_report_annotation_guidelines_final_with_affiliations_1.pdf

```
import torch_geometric.nn as gnn
import torch

class GCN(torch.nn.Module):
    def __init__(self, input_features_dim=300):
      super(GCN, self).__init__()
      self.conv1 = gnn.TransformerConv(input_features_dim, 128)
      self.conv2 = gnn.TransformerConv(128, 64)
      self.conv3 = gnn.TransformerConv(64, 64)
      self.classifier = torch.nn.Linear(64, 3)

    def forward(self, input):
      x = self.conv1(input.x, input.edge_index)
      x = self.conv2(x, input.edge_index)
      x = self.conv3(x, input.edge_index)
      classification_output = self.classifier(gnn.global_mean_pool(x, input.batch))
      result = {}
      result["classification_output"] = classification_output
      if input.y is not None:
        criterion = torch.nn.CrossEntropyLoss()
        classification_loss = criterion(classification_output, input.y)
        result["loss"] = classification_loss
      return result
```

Listing 3: Code definition of the GCN model

each word in the document using the word embedding technique (Bojanowski et al., 2016). We then create a graph with nodes representing the words and edges representing the co-occurrence relationships between them. Figure 1 depicts an example of a document represented as a graph.

Once we have created the graph structure, we apply graph convolutional operations to learn representations for the nodes in the graph. Graph convolutional operations involve aggregating information from a node's neighbors and updating the node's representation based on that information. This process is repeated multiple times to allow the network to learn increasingly complex representations of the nodes. Finally, we can use the learned node representations to classify the text document. This is typically done by applying a feed-forward neural network to the node representations to make a final classification decision. We used the PyTorch Geometric[2] library to train our GCN and the fasttext[3] library to compute word embedding. Listing 3 presents the code of our GCN model.

2.4 Sentence number-based model

The sentence number-based model leverages the distribution of sentence lengths in a text document to predict its genre. A Gaussian probability function is used to capture the shape of the distribution, and the area under the curve, for different numbers of sentences, is used to compute the classification probability. First, for each document class (satire, reporting, opinion), the number of sentences in a text document is determined. This can be done using a sentence tokenizer, which segments the text into individual sentences. Next, we use the mean and standard deviation of the sentence number of each class. Then, these statistics are used to fit a Gaussian probability function to the sentence number distribution. The Gaussian function can be expressed as follows:

$$f(x) = \frac{1}{\sqrt{2 \times \pi \times \sigma^2}} \times e^{\frac{-(x-\mu)^2}{2 \times \sigma^2}}$$

where x is the number of sentences within a document, μ is the mean sentence number, σ is the standard deviation of sentence number, and π is the mathematical constant. Figure 2 shows the Gaussian distribution of each genre class based on the computed sentence number.

```
from scipy.stats import norm

satire_norm = norm(loc=28.368421, scale
    =11.67243)
reporting_norm = norm(loc=29.337079,
    scale=17.319159)
opinion_norm = norm(loc=33.994832, scale
    =21.964172)


satire_prob = satire_norm.cdf(x)
reporting_prob = reporting_norm.cdf(x)
opinion_prob = opinion_norm.cdf(x)
```

Listing 4: Code definition of the GCN model

[2]https://pytorch-geometric.readthedocs.io/en/latest/
[3]https://fasttext.cc/

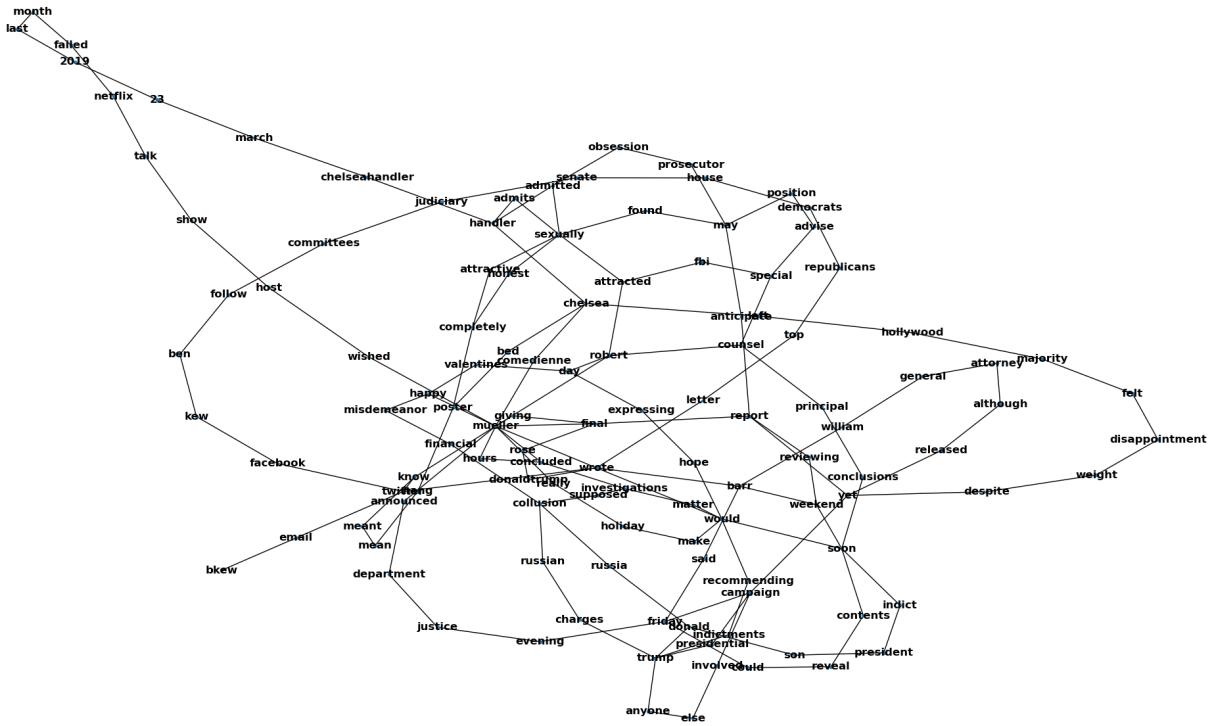

Figure 1: Graph representation of a text document

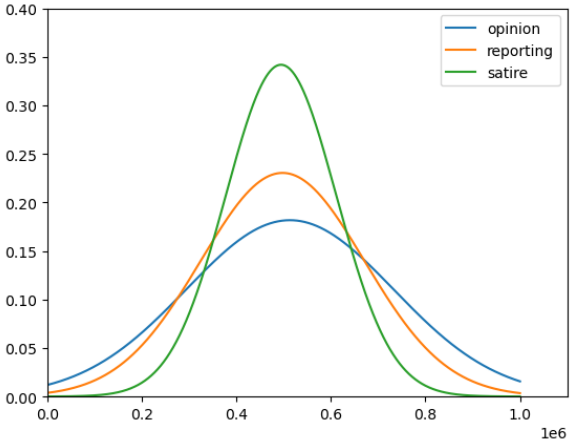

Figure 2: Gaussian distribution of each class (satire, reporting, opinion) based on the mean and standard deviation of sentence numbers

Finally, we use the area under the Gaussian function curve to compute the probability that a document belongs to a specific genre. While this model is relatively simple, it is effective for genre classification tasks, especially with the dataset given in subtask.

Listing 4 presents the code of the sentence number-based model. We combine the training set and validation set to compute the mean and standard deviation of the sentence numbers of each class.

2.5 Ensemble method

We use max voting to combine the predictions of the Longformer, RoBERTa, GCN, and sentence number-based models. It involves taking the class label that receives the most votes from the individual models as the final prediction. In case two models predicted the same label and it is different from the opinion, we select the predicted label as the final prediction.

3 Experimental results

3.1 Dataset

The organizers of SemEval 2023 task 3 provided English text documents annotated with their genre labels (Piskorski et al., 2023). The train set contains 433 documents where 88.22% (382 samples)

are opinions, 9.47% (41 samples) are reporting, and 2.31% (10 samples) are satires. The validation set consists of 83 documents where 65.06% (54 samples) are reporting, 24.10% (20 samples) are opinions, and 10.84 (9 samples) are satires. The test set contains 54 documents without labels. Figures 3 and 4 show the distribution of genre classes in the train set and the validation set. We applied the weighted random sampling technique to handle the imbalanced dataset problem, where the distribution of samples is balanced across classes during training. The idea is to sample the same number of elements from different classes in each batch during training.

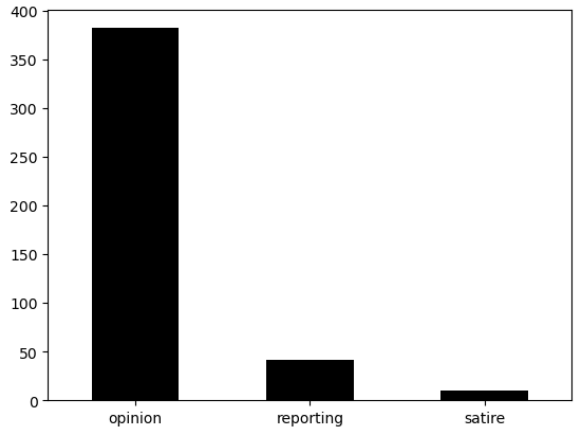

Figure 3: Distribution of genre classes in the training set

3.2 Experimental setup

We implemented our models using PyTorch (Paszke et al., 2019), Hugging Face (Wolf et al., 2020) and SciPy[4]. Table 1 presents the hyper-parameters we set to train our deep learning models. All hyper-parameters are chosen according to the evaluations performed on the validation set. We used the Adam optimizer in all experiments. All experiments are conducted using a Dell PowerEdge XE8545 server, having 2 AMD EPYC 7713 64-Core Processor 1.9GHz, 1TB of RAM, and 4 NVIDIA A100-SXM4-80GB GPUs.

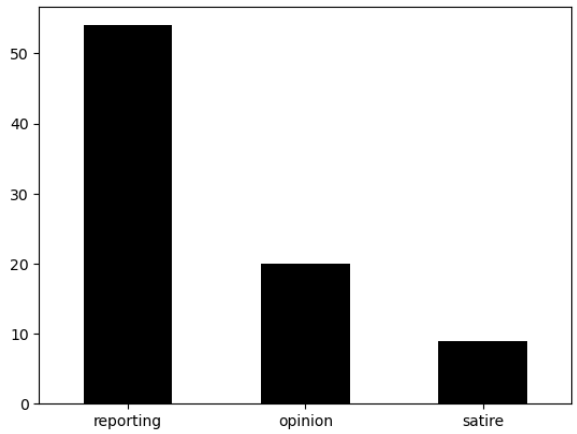

Figure 4: Distribution of genre classes in the validation set

Table 1: Hyper-parameters used to train deep learning-based models

Model	Train epochs	Learning rate	Weight decay
Longformer	9	2^{-5}	0.01
RoBERTa	9	2^{-5}	0.01
GCN	1000	-	0.0007

3.3 Performance evaluation

Table 2 presents the results obtained with our models in the genre classification task. The Longformer and RoBERTa models achieved the same scores and performed better than the GCN and sentence number-based models. The ensemble model improved the results by ∼7.4% F1 macro.

Table 2: Obtained results of the different models on the genre classification English test set

Model	F1 macro	F1 micro
Baseline	0.28802	0.61111
Longformer	0.31932	0.57407
RoBERTa	0.31932	0.57407
GCN	0.28802	0.61111
Sentence number-based model	0.17198	0.18519
Ensemble	0.39351	0.51852

4 Conclusion

This paper presents our method for English document genre classification in SemEval 2023 task 3, which leverages machine learning and deep learning techniques. Our method combines multiple predictors, including Longformer, RoBERTa, GCN, and sentence number-based model to predict the genre of an English text document. The obtained results show that our method achieved 39.35% F1 macro and ranked 12th in SemEval 2023 task 3. This demonstrates the effectiveness of our method in addressing news document genre classification tasks. We hope that our approach will inspire further research and improvements in the field of document genre classification.

[4]https://scipy.org/

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2016. Enriching word vectors with subword information. *CoRR*, abs/1607.04606.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Abdellah El Mekki, Ahmed Alami, Hamza Alami, Ahmed Khoumsi, and Ismail Berrada. 2020. Weighted combination of BERT and n-GRAM features for nuanced Arabic dialect identification. In *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, pages 268–274, Barcelona, Spain (Online). Association for Computational Linguistics.

Abdellah El Mekki, Abdelkader El Mahdaouy, Ismail Berrada, and Ahmed Khoumsi. 2022. Adasl: An unsupervised domain adaptation framework for arabic multi-dialectal sequence labeling. *Inf. Process. Manage.*, 59(4).

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Xien Liu, Xinxin You, Xiao Zhang, Ji Wu, and Ping Lv. 2020. Tensor graph convolutional networks for text classification. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8409–8416. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Jakub Piskorski, Nicolas Stefanovitch, Giovanni Da San Martino, and Preslav Nakov. 2023. Semeval-2023 task 3: Detecting the category, the framing, and the persuasion techniques in online news in a multi-lingual setup. In *Proceedings of the 17th International Workshop on Semantic Evaluation*, SemEval 2023, Toronto, Canada.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2018. Graph convolutional networks for text classification.