A comparative analysis of classical Machine Learning methods and a Transformer architecture for the classification of News articles

César Caramazana Zarzosa

I. Introduction

In this final project of Natural Language Processing we will be carrying a supervised classification task over a News dataset after applying three different text vectorization techniques –TF-IDF, Word Embeddings (W2V) and LDA Topic representation– and comparing the performance of classical Machine Learning methods versus a fine-tuned Transformer from the Hugging Face library.

The report is divided in four sections: in section II we will present the dataset and how the vector representation pipelines were implemented; in section III, the classifiers we used to carry out the experiments will be described, as well as the evaluation metrics to measure the performance and the results obtained; in section IV, the analysis of the results will be discussed; and in section V, the key findings and conclusions will be summarized.

II. Materials and methods

In this section we describe how the final dataset was generated from online resources and the specifics of the text vectorization techniques we applied.

A. The dataset

The source of the data is the News Category Dataset [1], a public dataset containing 210000 news headlines from 2012 to 2018 from the digital newspaper HuffPost [7], as well as a brief description, the author, the URL, the date and the label on the category of the piece of news. All the text is written in English.

We generated our final set as a subpartition of 16500 samples with the body of the articles retrieved from the original webpage. Each sample was processed with the following steps:

1) Valid Part-of-Speech (nouns, verbs, pronouns and adjectives) and alpha-numeric filtering.
2) Generic stopword removal.
3) Specific stopword removal, to eliminate a disclaimer that appeared in every article, *"By entering your email and clicking Sign Up, you're agreeing to let us send you customized marketing messages about us and our advertising partners"*, and other undesirable terms from the HTML that spoilt the topic modeling task.
4) Tokenization: with lemmatization and to lower case.
5) Bi-grams detection: to merge words that commonly appear together in the data, such as "New York", "North Korea" or "White House".

The average number of words per article is 213.97, distributed as shown in Fig. 1, following a quasi-Gaussian distribution with just very few samples surpassing the 1000 words mark.

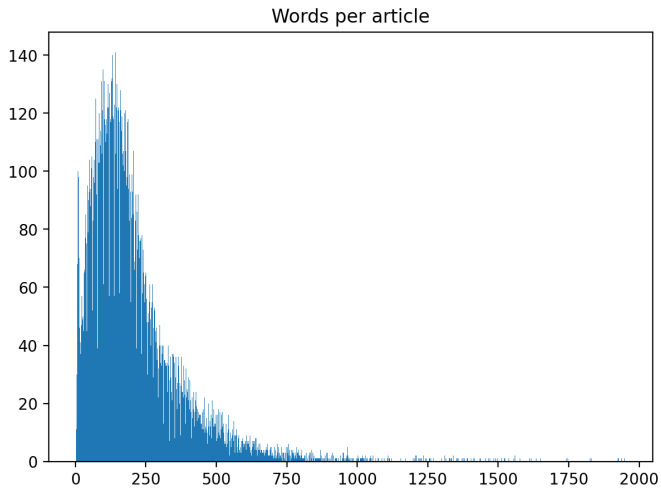

Fig. 1. Histogram of number of words per article. For a better visualization, documents with more than 2000 words (16 out of the 16500) where filtered out.

Regarding the annotations, while the original data had 42 categories –such as POLITICS, BUSINESS, SPORTS, LATIN VOICES, EDUCATION or CRIME–, the subpartition we used presented only 35, very unevenly distributed, as is shown in Fig. 2. To simplify the classification task, we selected POLITICS, WORLD NEWS and ENTERTAINMENT as stand alone classes and merged everything else into an OTHER class, which resulted in a distribution that is shown in Fig. 3.

The Huffpost is an American newspaper, so all their coverage is biased towards the reality of the United States.

B. Vector representation

1) TF-IDF: The TF-IDF matrix was computed automatically using Gensim's implementation. This method returns a very sparse representation of each document, for a vocabulary size of 21184 words.

2) Word embeddings: We decided to train a Fast Text model [3] with vector size 200 and window size 5 to be robust against out-of-vocabulary words. The document embedding V is then calculated averaging the embeddings of the words in the document weighted by their tf-idf factor. With this strategy we control the contribution of common versus uncommon words in the embedding space.

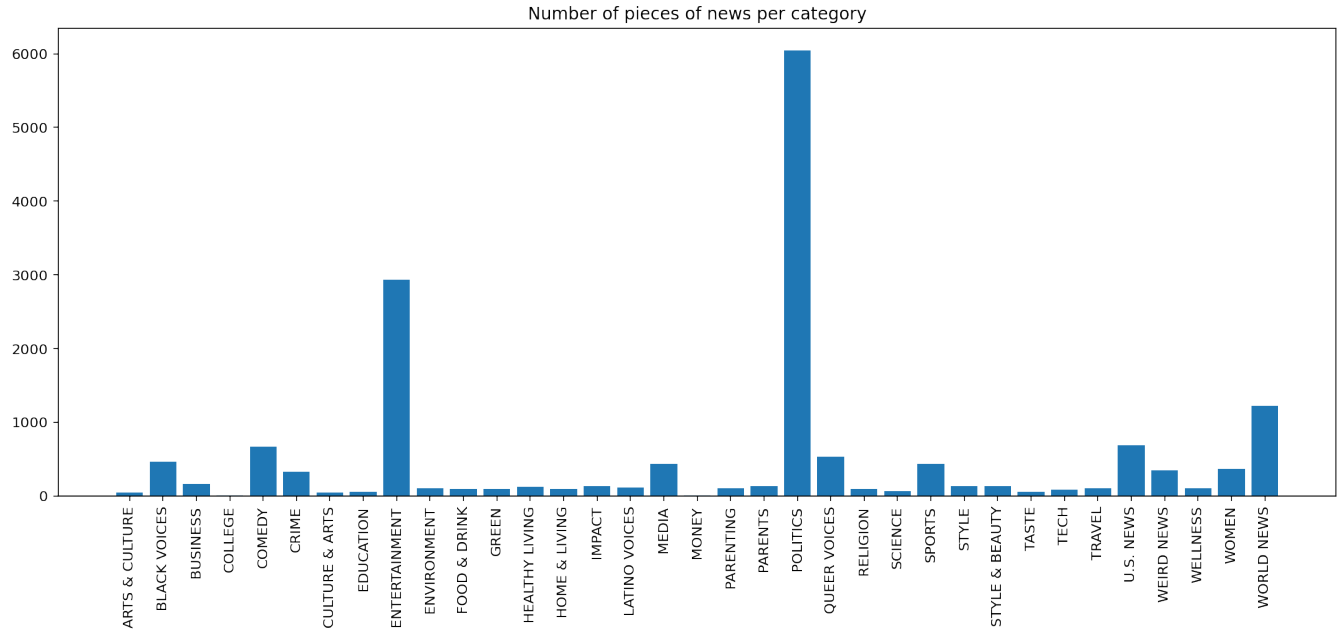

Fig. 2. Histogram of number of samples per labeled category in the subpartition of 16500 articles.

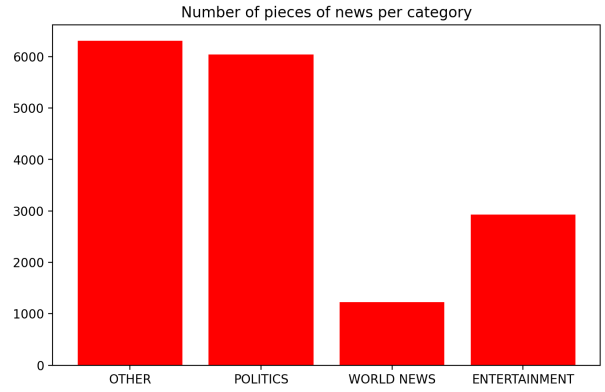

Fig. 3. Histogram of number of samples per class.

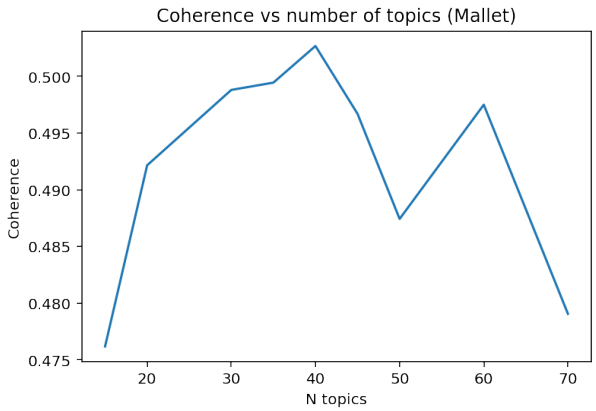

Fig. 4. Coherence versus number of topics in LDA topic modeling (with Mallet).

$$V(doc) = \frac{1}{\#words} \sum_{word} tfidf(word, doc) V(word) \quad (1)$$

3) LDA Topic representation: The LDA topic representation was carried out using Mallet [4] and in two steps. First, an exploratory phase to search for the number of topics with the highest coherence score and to inspect the resulting words per topic. This phase was iterated over several times, preprocessing the text after inspecting the outcome to obtain better coherence (mainly stopword removal, common-words filtering and bi-grams detection). Second, a re-training of the best model (n = 40) for a greater number of iterations. The curve of the exploratory phase is shown in Fig. 4.

Some of the most intuitive resulting topics relate to particular world events that happened in the past few years, such as the U.S. presidential elections of 2016, the Russia-Ukraine war or the Covid pandemic. Other topics are broader and more generic, such as Climate Change, Religion or Racism. A few ended up being too noisy to be associated to a particular thematic or occurrence, composed of common terms that appear in many documents, even after applying the specific stop-word removal and increasing the aggressiveness of the dictionary filtering. In general, although the topics cannot be linked directly to the annotated categories of the dataset, they share a resemblance (e.g., the topics related to Religion or Racism have a correspondence with the categories RELIGION and BLACK VOICES). In the attached notebook you may find a graphical visualization of the topics using LDAvis [5].

III. Experimental set up and results

In this section we will describe the experimental set up and display the final results.

A. Machine Learning models

1) Classical methods.: Three classification paradigms were chosen from the Scikit-learn library to serve as the baseline of classical Machine Learning methods. These are: Support Vector Machines (SVM), Random Forest classifiers (RF) and Multilayer Perceptrons (MLP).

- SVM: with a RBF kernel. The parameter C was cross-validated.
- RF: the number of trees and their depths were cross-validated.
- MLP: with 3 hidden layers of sizes 128, 256 and 128. The Adam algorithm was used to optimize the weights. Training was carried out for 1000 iterations with early stopping.

The hyperparameters of each approach were selected via grid search with a 5 fold cross validation process. While we attempted to keep the grid of the search equal for every vector representation, the training of the SVM with the TF-IDF matrix was too costly and the range of values for the parameter C was reduced with respect to the W2V and the LDA approaches.

2) Transformer: Regarding the Transformer architecture, a pretrained DistilBERT model [6] was fine-tuned for 5 epochs. Some minor tweaks had to be done to accommodate for our multi-label classification task. The appropriate data preprocessing was also carried out to fit the required format of the model (such as one-hot encoding the labels and tokenizing the inputs).

B. Evaluation metrics

In order to obtain statistically significant results, the dataset was divided into a train and a test set, in a 65/35 proportion, that is, 10724 and 5775 samples respectively. The split was stratified to maintain the same proportion of classes. For the Transformer, the train was further divided into train and validation.

The performance of the models is evaluated in terms of accuracy. For a better disclosure of signs of overfitting, it is calculated over both the training and the test sets. In some key cases, the confusion matrix is also reported. We also compute the average inference time (in seconds per sample), to further analyze the computational cost of each approach.

C. Results

In Table I the complete summary of the results is shown[1].

In Fig. 5, two plots of the confusion matrices for the worst and best performing models (in terms of test accuracy).

IV. Discussion

In this section we discuss the results and performance of each approach, providing a comparative analysis on both the ML methods and the vector representations. For an easier identification of the experiments, we will use the notation ML-method/Vector-representation to name the corresponding row/column of Table I. For example, SVM/TF-IDF refers to the Support Vector Machine trained with the TF-IDF matrix representation. Unless stated otherwise, the reported accuracy values in this section are those calculated over the test set.

On the data. First of all, it is important to notice the difficulties that arise from the data, namely, that the labeled categories are very broad, and therefore present some ambiguity regarding the class an article may belong to, and that the dataset is imbalanced. As our problem is multilabel but not multiclass, this ambiguity may be critical to correctly classify articles that cover an "in-between" topic (e.g., news related to the foreign policy of the U.S. that can be considered either POLITICS or WORLD NEWS). Given that the final dataset generation, as a partition of the original source, was done without taking into consideration the labels, we had to face a heavily imbalanced problem, even after the merging of classes. Regarding the dataset size, the results indicate that 16.5k samples were more than enough to carry out the task where, having the retrieval and preprocessing pipeline implemented, the cost of adding more samples is mainly time.

On the performance (ML methods). The worst performing combination overall was the RF/TF-IDF. To explain why this is the case we need to understand how this kind of classifier makes a decision. A Random Forest is an ensemble of many decision trees that, at each node, splits the input space in two by applying a threshold on the features with more discriminative power (Gini index). In TF-IDF, each feature is associated with a word from the vocabulary and, therefore, the splits are done considering the appearance frequency of particular terms in each of the categories. For example, let's say that in an individual tree the first node checks if the TF-IDF factor associated with the token "Trump" is greater or lesser than 0.3; and in the two resulting branches, it is checked if the TF-IDF factor of the tokens "movie" and "Korea" are greater or lesser than 0.15. The four possible combinations of the conditionals end up determining the class. All the trees follow the same procedure, with other tokens, and the final label is obtained through a majority vote. As the input matrix is very sparse (that is, most of the elements are zero), these approach struggles to generalize to all the documents. Even if each tree pays attention to a different combination of words, for many samples the TF-IDF values will be zero, and in these cases the vote will be proposed in absence of information. Furthermore, since we established a maximum depth per tree of 3 –because ensembles are usually a combination of weak classifiers–, and cross validated the number of trees up to 100, the Random Forest is not expressive enough to solve the problem —as mentioned, each tree considers only 4 words–. Indeed, as we can see in Fig. 5, the RF/TFIDF has become a binary classifier between OTHER or POLITICS (no samples are classified as ENTERTAINMENT or WORLD NEWS). The RF/W2V and RF/LDA are also affected by the short-comings of the classifier, although the performance is better than with

[1]As for the Transformer the training set was split into train and validation, the accuracy value reported on Train for this approach is computed over 80% of the samples that were used for the rest of the models

		TF-IDF	W2V	LDA	Tokens
SVM	Train	0.999	0.805	0.842	
	Test	0.799	0.765	0.771	
	Inference time	9.3·1e-3	2.0·1e-3	7.5·1e-4	
Random Forest	Train	0.587	0.679	0.650	
	Test	0.581	0.671	0.638	
	Inference time	3.4·1e-5	7.6·1e-6	1.2·1e-5	
MLP	Train	0.955	0.798	0.795	
	Test	0.798	0.765	0.767	
	Inference time	4.5·1e-5	1.6·1e-5	1.6·1e-5	
Transformer	Train				0.964
	Test				0.829
	Inference time				1.7·1e-2

TABLE I

RESULTS FOR EACH MACHINE LEARNING METHOD (ROWS) AND EACH TEXT REPRESENTATION (COLUMNS), IN TERMS OF ACCURACY (TRAIN AND TEST SETS) AND AVERAGE INFERENCE TIME (IN [S/SAMPLE]).

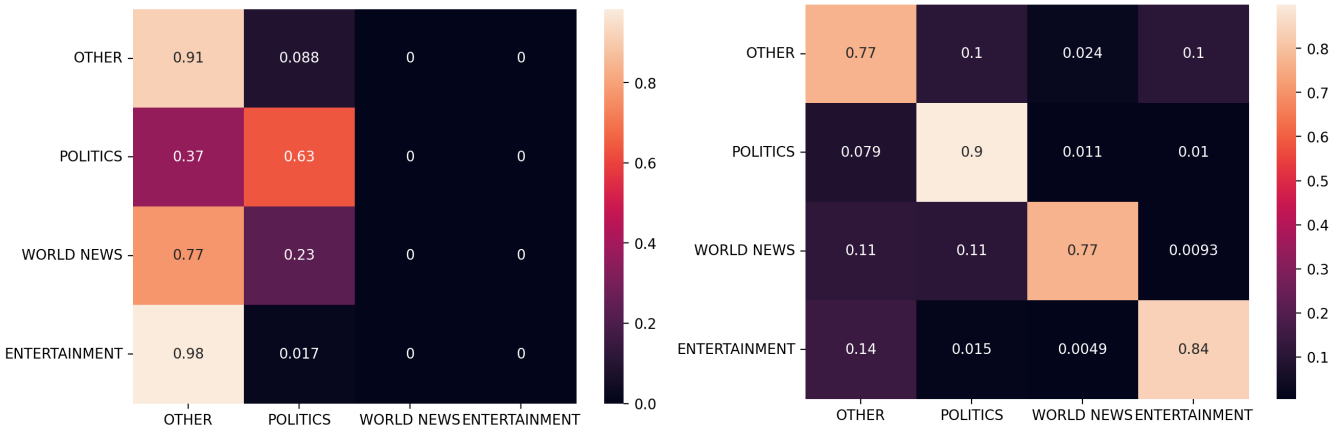

Fig. 5. Left: confusion matrix of the Random Forest trained with TF-IDF. Right: confusion matrix of the Transformer. (Test set)

TF-IDF because each component on the embedding and on the topic vector contains information from more than just one word (more contextual information). All these reasons explain why the Random Forest was not a suitable option for our task, returning the poorest results.

The best performance was yielded by the Transformer, with a 0.828 accuracy and a breach of 0.13 between test and train (overfitting). The main disadvantage is the computational cost, as it is the approach that takes the longest inference time per sample –four orders of magnitude higher than the fastest (RF/W2V) and one order of magnitude higher than the second slowest models (SVMs)–. Nevertheless, we need to take into account that the architecture is much more complex and the number of parameters to be optimized is much higher than any other. In the confusion matrix (see Fig. 5, right) we can see that the errors come from mistaking the stand-alone categories with OTHER, on the one hand, due to the ambiguity of the data we mentioned, and on the other hand, due to the bias introduced by the fact that it is the most represented class (highest a priori probability). The most errors are produced between ENTERTAINMENT and OTHER, likely caused by thematic overlapping.

The SVMs and MLPs offer very similar capabilities, closer to the Transformer's than to the Random Forest's. The main difference between them is again the inference time: longer for the former despite keeping the number of hidden layers to 3 (not a very deep configuration). In both, the best accuracy was achieved with TF-IDF, although it is the case with the most worrisome signs of overfitting, as they are clearly memorizing the training set and achieving an almost perfect score.

On the performance (Vector representations). Between W2V and LDA there are not significant differences for any of the ML methods. Even though the techniques are very distinct, with both we are compressing the global information of the article in a low dimensional vector (sizes 200 and 40 respectively), which seems to capture to a greater o lesser degree the theme of the pieces of news and help discriminate the classes from each other. A priori we could have guessed that the topic representation would be a better fit to our task,

given that the labeled categories of the dataset are already topics themselves. While this is still true, the results teach us that other techniques can be equally useful, and even outperform LDA.

However, a disadvantage of LDA in comparison with TF-IDF or W2V is that we had to set the number of topics before the classification task was carried out. Maybe selecting a number of topics equal to 4 would result in a representation that aligns better with the 4 categories we aimed to classify the documents in, but the criteria and the search range we chose (maximum coherence value) pushed the final number up to 40. We can only speculate if coherence should be switched to accuracy, cross-validating the number of topics as a model hyperparameter. This idea, of course, would have increased severely the cost of the whole pipeline and, given the time constraints of the project, the option was discarded.

The Fast Text model could have also be trained with various vector sizes, but the impact on the classification task should be minor. An advantage of the W2V approach is that we could easily add more articles to the corpus and we wouldn't need to re-train the embeddings model, something we would need to do with TF-IDF and LDA.

With the Transformer we did not have to worry about any of this as it takes the raw text as input.

While count-based methods, such as TFIDF, do not consider neither syntactic nor semantic dependencies among words, for our particular task it did not imply a significant drop in performance. Actually, from the results we could say that the ML methods have a bigger impact on the final accuracy than the vector representations. With the SVM and the MLP, TFIDF overfits, but not W2V or LDA. The Transformer also overfits, but we cannot say if its due to the input or to the architecture itself.

On the computational cost. As we would expect, the most expensive vector representation is the TF-IDF. Despite the fact that the vectors are sparse (and the latest versions of Scikit-learn allow sparse matrices to be input to the models we tried out), the average dimensionality of the articles, that is, 213.97 words (see Fig. 1), is bigger than the W2V embedding and the topic representation, which have a fixed length for all the samples –of course this was a design choice as we could have reduced the dictionary to a much smaller vocabulary–. As a consequence, TF-IDF does not scale well with the vocabulary size nor with the number of words in a document, which can be considered a critical drawback that is not present in W2V and LDA. For the volume of our dataset, these differences in time haven't been extremely problematic, although it compromised the cross-validation process of some models (specially RF/TF-IDF, as mentioned before). On the other hand, the Transformer also requires a fixed input size to be able to batch the samples, which is done via padding. This means that in many cases the input is larger than strictly necessary –this is, however, an intrinsic handicap of neural networks since they cannot handle variable input sizes as we would ideally prefer for natural language processing–. The Transformer was the method that simultaneously presented the highest accuracy and the highest computational cost. A better trade-off was obtained with the MLP/TF-IDF, which yielded the third best accuracy (almost the same as the SVM/TF-IDF, which scored second) with much lower inference time.

We did not contrast the cost of the training processes, primarily due to the fact that the cross-validation of the specific parameters of each method is not comparable –the grid search could be arbitrarily increased or decreased–.

V. Conclusions and further work

To summarize our findings: out of all the classical Machine Learning models, the Random Forest classifiers provided the worst performance overall, particularly for the TF-IDF input, due to the lack of expressiveness. The Support Vector Machines and the MLPs yielded similar results, and scored a bit lower than the DistilBERT fine-tuned Transformer, with which the best test accuracy of 0.829 was achieved. We analyzed the short-comings of each vector representation, such as the large dimensionality of the TF-IDF input matrix or the need to set beforehand the number of topics for the LDA model. A comparison of the computational cost was carried out in terms of average inference time: the Transformer was the most costly method due to the complexity of the architecture while the classical ML classifiers were much more light-weighted. We discovered that many errors in the classification are related to the ambiguity and the class-imbalance of the data.

Finally, we found out that the Machine Learning method had a greater impact in performance than the text vectorization technique, although it is the latter that influenced the most on the computational cost of the inference process, as the input dimensionality is very heterogeneous for the different representations.

Some possible lines of improvement involve: the cross-validation of the number of topics in LDA as a hyperparamenter of the classification pipeline, a finer search of the threshold for the detection of N-grams or the deepening in other Transformer architectures, as they outperform the classical approaches we took in this project.

References

[1] Rishabh Misra. "News Category Dataset". In: *arXiv preprint arXiv:2209.11429* (2022).

[2] Rishabh Misra and Jigyasa Grover. *Sculpting Data for ML: The first act of Machine Learning*. Jan. 2021. ISBN: 9798585463570.

[3] Piotr Bojanowski et al. "Enriching Word Vectors with Subword Information". In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146. ISSN: 2307-387X.

[4] Andrew Kachites McCallum. "MALLET: A Machine Learning for Language Toolkit". http://mallet.cs.umass.edu. 2002.

[5] Carson Sievert and Kenneth Shirley. "LDAvis: A method for visualizing and interpreting topics". In: *Proceedings of the workshop on interactive language learning, visualization, and interfaces*. 2014, pp. 63–70.

[6] Victor Sanh et al. “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”. In: *ArXiv* abs/1910.01108 (2019).

[7] *Huffpost*. URL: https://www.huffpost.com/.