Team INSA Passau at Touché: Multi-lingual Parliamentary Speech Classification

Notebook for the Touché Lab at CLEF 2024

Maud Andruszak[1,2], Alaa Alhamzeh[2], Előd Egyed-Zsigmond[1], Anton Carlsson[1], Johan Leydet[1] and Yasser Otiefy[2]

[1]*INSA de Lyon, 20 Avenue Albert Einstein, 69100 Villeurbanne, France*
[2]*Universität Passau, Innstraße 41, 94032 Passau, Germany*

Abstract

In this paper, we present the architecture used for our participation in the shared task of Ideology and Power Identification in Parliamentary Debates - Touché Lab at CLEF 2024. This task aims to identify the ideology of the speaker's party, and to identify whether the speaker's party is currently governing or in opposition. Furthermore, the data associated with these two sub-tasks are proposed from a multilingual perspective, where the speeches belong to at least 29 national or regional parliaments. Among our submitted runs, we achieved the best performance through BERT fine-tuning for both sub-tasks, in addition to Llama 3 prompting for a subset of the parliaments on identifying the speaker's party.

Keywords

Political debates, Large language models (LLMs), Few shot learning, Text classification

1. Introduction

The identification of ideology and power structures in parliamentary debates is of vital importance for a comprehensive understanding of political dynamics and decision-making processes. Recognizing the background of ideological stances and power relations within debates facilitates a deeper insight into the strategic maneuvers of policymakers and the potential biases influencing legislative outcomes. Given the extensive volume of debates and the complexity of political discourse, the need to automate this identification process is increasingly urgent. Automation not only enhances the efficiency of analysis, but it may also ensure a more objective and consistent evaluation of the data, devoid of human biases.

The current edition of the Touché shared task [1] tackles these issues by suggesting two sub-tasks:
Sub-Task 1: Given a parliamentary speech in one of several languages, identify the ideology of the speaker's party.
Sub-Task 2: Given a parliamentary speech in one of several languages, identify whether the speaker's party is currently governing or in opposition.

We have participated in both sub-tasks as team *INSA*.

The baseline defined by the organizers of the task is a linear logistic regression, using the term frequency-inverse document frequency (TF-IDF) to process the texts. The results using this baseline differ a lot between the parliaments, but also from the same parliament between the two sub-tasks. This entails that some approaches may perform better for some parliaments/sub-tasks than another. In addition, Alhamzeh et al. showed in an intensive empirical study [2] on text classification tasks, that some classical machine learning methods (specifically SVM) can outperform more complex deep learning models (BERT in their case). Hence, we decided to study varied methods.

Briefly, we have implemented four approaches. Two of them are based on Large Language Models (LLMs): the first one consists of fine-tuning a LLM, which we did with BERT as well as Llama 3, and the second one is based on prompting an LLM. Our third approach is based on manual features and is

CLEF 2024: Conference and Labs of the Evaluation Forum, September 09–12, 2024, Grenoble, France
maud.andruszak@insa-lyon.fr (M. Andruszak); alaa.alhamzeh@uni-passau.de (A. Alhamzeh); elod.egyed-zsigmond@insa-lyon.fr (E. Egyed-Zsigmond); anton.c1998@gmail.com (A. Carlsson); johan.leydet@insa-lyon.fr (J. Leydet); yasser.otiefy@uni-passau.de (Y. Otiefy)

linked to the frequency of discriminatory words in a text, and the last one is a Support Vector Machine (SVM).

This paper is organized as follows: All of our submitted approaches will be presented in Section 2. We elaborate on the outcomes for both subtasks and different parliaments in Section 3. Finally, we conclude our work in Section 4.

2. Methods

2.1. Fine —–tuning pre-trained LLM

Our first approach consists of fine tuning a Large Language Model (LLM), which is an advanced artificial intelligence system designed to understand and generate human language. Utilizing deep learning and neural network architectures like transformers, these models are trained on extensive text datasets, allowing them to predict and generate coherent text based on context. LLMs excel in various tasks such as text generation, translation, summarizing, question answering, and sentiment analysis, making them invaluable for applications like chatbots, content creation, education, and medical diagnosis. They represent a remarkable leap in enabling sophisticated human-machine language interactions.

To train and test our models, we split the train dataset in two parts: 80% for training and 20% for testing. To match the task evaluation strategy, we evaluated our runs using the macro-averaged F1 score, which calculates the harmonic mean of precision and recall while assigning equal weight to all classes.

2.1.1. BERT

Among available open source LLMs, we examined the Bidirectional Encoder Representations from Transformers (BERT) [3]. BERT is a transformer that can be fine-tuned by adding an output layer. On Hugging-Face,[1] we can find a lot of transformers based on BERT. We looked at several of them, trying to find the one that fitted our task the best. We studied the following transformers, some of them general like bert-base-cased, bert-base-uncased, roberta-base, and some others trained on law data like nlpaueb/legal-bert-base-uncased [4], casehold/legalbert [5], saibo/legal-roberta-base[2] or pile-of-law/legalbert-large-1.7M-2 [6]. All of these are trained to support English written texts. Consequently, we tested those transformers on data from the British parliament (GB).

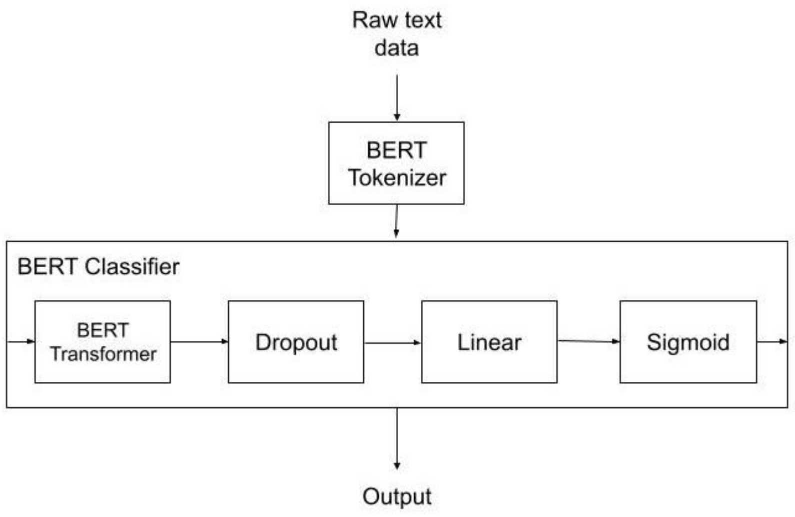

Figure 1: Overview of our BERT fine tuning model

[1]https://huggingface.co
[2]https://huggingface.co/saibo/legal-roberta-base

Our system for this approach consists of five steps, as shown in Figure 1. First, we use the tokenizer corresponding to our LLM transformer to tokenize the input texts. Then, our BERT classifier takes these new inputs to be applied on the LLM transformer. After this, the transformer output goes through a dropout layer, a linear transformation with an output of dimension 1 and finally a sigmoid layer. The dropout layer consists of cutting nodes from the input and hidden layers of the neural network, which avoids overfitting [7]. The sigmoid layer maps the input to a floating point between 0 and 1, which is easily suitable for binary classification [8]. This floating point is our final result. The hard label is computed with a simple threshold set at 0.5, as it is on the task baseline.

The experiment was carried out with 3 folds cross-validation. The parameters searched for optimization were the LLM transformer as well as the dropout probability, the number of epochs, the learning rate and the tokens length. We ended up taking the bert-base-uncased transformer, which showed the best F1 score on this experiment. Our dropout probability was set to 0.2, the number of epochs to 3, the learning rate to 0.00001, and the tokens length to 150. As for the second step, we optimized the system by choosing the best loss function. We tested the cross-entropy loss and the binary cross-entropy loss. The latter was the best. With all these parameters, we created a baseline with BERT for every parliament, by training each of them separately on their English translations.

With our BERT baseline results, we noticed that not all parliament texts behave the same when put inside a BERT model. We have improvements of the F1 score for some countries, but not for all of them. Then, we focused on improving the results by augmenting the training data. To do so, we firstly trained a single model on the English translations of every parliament at the same time. After this, we tried to increase the F1 score while having the minimal best training data, meaning that we may not need to add data from all parliaments to have better results. We did this with two things in mind : energy consumption, as well as time efficiency. Indeed, less training data decreases the time needed to train the model, and some parliaments might not help in a certain parliament predictions. That’s why we implemented an incremental process to add a parliament’s data to the training data until we decide that adding some data is useless, starting from the sole studied parliament’s data. For this, we trained every pair of parliaments of a same sub-task. Then, for each parliament, we sorted the pairs containing this parliament in descending order of F1 score. After that, we used this order to incrementally add a parliament to the training, each time adding the one from the pair with the best F1 score decrementing. As explained above, we stopped the incremental process when it was not helping anymore.

2.1.2. Llama 3

Llama 3 [9] is the latest model of Meta’s LLM series so far/to date. It significantly enhances the capabilities introduced by its predecessors. This model was released on April 18, 2024 and comes in multiple configurations, including 8 billion and 70 billion parameters, optimized for both performance and scalability. Notably, Llama 3 has been trained on an extensive dataset of up to 15 trillion tokens, demonstrating improved performance even with smaller, more efficient models. This efficiency allows it to generate high-quality results comparable to larger models while being easier to deploy and run on standard hardware configurations.

We only tested Llama 3 on the GB parliament data of sub-task 1 (orientation). We chose this parliament because the original language of this dataset is English, and Llama 3 is trained mainly on English-speaking data. The hyperparameter tuning step was realized with a grid search on a 5 folds cross-validation and resulted with this configuration:

- Model name: "meta-llama/Meta-Llama-3-8B"
- Number of epochs: 5
- Learning rate: 0.00005
- Maximum tokens length: 256

This configuration yielded a mean F1 score of 0.686.

We could not find enough time to train and test on other parliaments. Even if the mean F1 score is not really high, we decided to submit this run with the best configuration of this approach.

2.2. Prompting

Prompt engineering is the process of crafting and refining the inputs given to a large language model (LLM) to achieve desired outputs. This technique involves carefully designing the phrasing, structure, and context of prompts to guide the model's responses effectively. By manipulating prompts, users can optimize the performance of LLMs in various applications, such as generating specific types of content, answering questions accurately, or performing complex tasks. Prompt engineering requires a deep understanding of how LLMs interpret and process language, enabling users to harness the full potential of these models while minimizing errors and biases in their outputs. This practice is crucial for maximizing the utility and accuracy of LLMs in real-world scenarios.

For the sub-task 1 (orientation), we are employing Llama 3, the latest generative text LLM made by Meta. The approach focuses on prompt engineering to guide the model to the specific task at hand.

The base prompt is composed of three distinct parts: the definition of the task, the sentence that we want to classify and the expected labels.

Here are the specific strategies we implemented:

- **Zero-shot learning**: We provide the model with the classification task without any prior examples, relying on its pre-trained knowledge to make accurate predictions. The prompt used is the following:

  > "Given this parliamentary speech '[text]', identify if the speaker's ideological party is left or right. Answer just with one word either 'Left' or 'Right'"

- **Few-shot learning** with different examples: We experimented with two-shot learning, providing the model with two examples to improve its classification performance:
  - **Incorrect predictions**: Using two examples where the model initially predicted incorrectly in the zero-shot experiments. We are using one example for each label (left and right).
  - **Mixed correctness**: Using one correct and one incorrect example, again with one for each label.
  - **Highest cosine similarity examples**: Selecting examples with the highest cosine similarity to all other examples, to provide contextually relevant training data.

  The prompts used in this strategies are like this:

  > "Identify if the speaker's ideological party is left or right. Answer just with one word either 'Left' or 'Right' only for the last text. Text: '[text from left]', Answer: 'Left'. Text: '[text from right]', Answer: 'Right'. Text: '[text to classify]'"

- **Voting approach**: We combine the results from multiple prompt variations to make a final classification decision based on the majority vote, thereby improving the robustness of the predictions. In this voting method, we use three predictions: the zero-shot learning prediction and two different few-shot learning prediction with mixed correctness.
- **Zero-shot learning with labels explained**: We enhance the zero-shot prompts by including an explanatory sentence about the political labels. This sentence clarifies what "left" and "right" typically refer to, as follows:

  > "Usually, Left advocates for social equality through government intervention and prioritizes issues like economic redistribution and social justice, while Right emphasizes individual liberty, free market principles, and traditional values, often favoring limited government intervention and policies that promote economic freedom."

As we have aforementioned, there are two variants of Llama 3 model: either with 8 billion parameters (8b) or with 70 billion parameters (70b). We have tested the zero-shot learning approach on both of them, and immediately realized the result differences between them. We tested those on 20% of the GB orientation dataset, that is to say 4 848 rows. We respectively obtained F1 scores of 0.758 and 0.811 with 8b and 70b. By testing these strategies on part of our training data, we determined the ones that could be useful. Thereby, we eliminated the few-shot learning strategy using incorrect predictions, as it was influencing the answer too much given the fact that every example was going in the opposite direction as what the model wants to answer. We also eliminated the few-shot learning strategy using the highest cosine similarity examples as the results were significantly worse than the zero-shot learning strategy. It seems that the examples with high similarity are not helping the model, we may just interfere in the prediction because of their length : these examples are 4 to 8 times longer than the average (2540 characters). This means they are not more discriminatory, only longer so they contain more words and thus have more similarity to the other texts.

The results of all our strategies kept for this approach, evaluated with the 70 billion parameters configuration[3], can be found in Table 1. To balance the two different runs with the few-shot learning strategies, we put an example correctly predicted 'Left' and one labelled 'Right' but wrongly predicted 'Left' on the first one, and the other way around for the second run.

Table 1
Llama-3-70b prompting results on sub-task 1 from British training data (GB)

Strategy	F1 score	Precision	Recall
zero-shot learning	0.811	0.83	0.814
few-shot learning with mixed correctness 1	0.791	0.813	0.794
few-shot learning with mixed correctness 2	0.799	0.814	0.802
voting	0.805	0.824	0.808
zero-shot learning with labels explained	0.776	0.807	0.781

It seems also important to notice that we can achieve decent results using this prompting method on the sub-task 1 (orientation), but it appeared to be much more complicated to gain anything from this approach on the sub-task 2 (power).

By employing these prompt engineering techniques, we aimed to leverage Llama 3's generative capabilities to improve the accuracy and reliability of political orientation classification. The careful design and testing of various prompt strategies allowed us to explore different aspects of the model's understanding and adaptability to the task at hand.

2.3. Manual Features

Besides LLMs, we have examined a more fundamental approach to the problem that is based on finding manual features, certain basic features of the texts that might be useful when classifying. Each manual feature results in one value for each text, which can then be used either as a complement to the prediction of the LLM, or as a prediction on its own.

2.3.1. Z-score

If given a corpus with two distinct parts, P_0 and P_1, it is possible to calculate how over- and underused each lexical token in P_0 is in comparison to P_1. The formula used is as follows:

$$\mathrm{Z-score}(t_{ij}) = \frac{tf_{ij} - n_j \cdot p(t_i)}{\sqrt{n_j \cdot p(t_i) \cdot (1 - p(t_i))}}$$

An explanation of the terms can be found in Table 2. A high z-score for token t_{i0} indicates that token i is used more frequently than expected in corpus P_0 compared to corpus P_1, and a low z-score indicates that the token is used less frequently in corpus P_0 than expected [10].

[3]https://huggingface.co/meta-llama/Meta-Llama-3-70B

Table 2
Explanation of the terms used when calculating the Z-score. Index j represents the corpus, and can have value either 0 or 1. Index i represents the index of the current token, and can take values from 1 to N, with N being the number of tokens in the corpus.

Term	Explanation
t_{ij}	Token i, present in corpus j
tf_{ij}	Occurrence frequency of token i in corpus j
n_j	Total number of tokens in corpus j
$p(t_i)$	Probability of token i being selected when randomly sampling both corpora, estimated as $(tf_{i0} + tf_{i1})/n$

2.3.2. Running Z-score sum

Given that the data for both the orientation and the power task is divided into two separate labels, 0 and 1, it is possible to calculate the most over- and underused tokens for each label for each country. This was done from the point of view of label 0, meaning that a high z-score signifies that a token is overused in texts with label 0, and a low z-score signifies that a token is underused in texts with label 0. For the corpus of each country, the z-score for each token was calculated. All tokens where the absolute amount of the z-score was above 2 were deemed discriminatory and thus usable to identify the label of a given text, but at most 5% of the total tokens for a country's corpus were allowed to be classified as discriminatory. Different discriminatory words were used for the orientation and power task for the same country.

As an example, we present in Table 3 a few of the most discriminatory words from the Swedish dataset. Interestingly, they indicate that topics such as immigration, law and order, and economics are more frequently discussed by the right-wing parties, as indicated by the overuse by words such as *migration, immigration, finance, police*, and *entrepreneurs.* The left-wing parties on the other hand seem to focus their debates on equality, climate, and social benefits, since the overused words include *climate, welfare, women, sustainable*, and *racism.* Another interesting note is that *bourgeois* is included in the words overused by the left-wing parties. The term *bourgeois* has traditionally been used by the left in Sweden to address the right.

Table 3
A few of the most discriminatory words for the Swedish dataset. The two leftmost columns have words with z-scores less than zero, which are overused in datapoints with label 1. The two rightmost columns have z-scores greater than zero, which are overused in datapoints with label 0.

Word	Z-score		Word	Z-score
Minister	-30.73		Work	16.07
Center	-16.87		Bourgeois	15.46
Liberals	-12.38		Climate	11.93
Christian	-11.48		Welfare	10.50
Migration	-11.19		Women	10.30
Immigration	-10.91		Sustainable	10.20
Finance	-10.62		Human	10.06
Police	-10.56		Society	9.89
Democrats	-9.58		Racism	9.74
Entrepreneurs	-8.69		Left	9.57

For each text, two manual features were calculated using the discriminatory words:

1. For each token that was considered discriminatory, its number of occurrences were added to a running sum if the z-score of the word was greater than zero, or subtracted from the running sum if the z-score was lower than zero.

2. For each token that was considered discriminatory, its z-score was multiplied by the number of occurrences, and that value was added to a running sum.

For both metrics, all the values were normalized to have a mean of zero and a standard deviation of 1. This was done country-wise.

To classify a given text, the optimal decision boundary was calculated. This was done by sliding the decision boundary from the lowest recorded running sum up to the highest recorded running sum, and for each decision boundary classifying all texts with a lower running sum as 1 and with a higher running sum as 0. The optimal decision boundary was considered the one that resulted in the highest F1 score. The datasets for some countries were very uneven, with upwards of 80% of the texts belonging to one label. To make sure that the optimal decision boundary was not just classifying everything as the majority label, for each country that had a label split of more than 60/40, texts from the majority label were randomly removed until a 50/50 split was achieved.

Since there were two different ways of calculating the running sum, for each language the one that resulted in the best F1 score was kept. Thus, for a given text from a given country, a prediction was made as follows:

1. Calculate the running z-score sum according to the best method for the country
2. Normalize the sum with the mean and standard deviation calculated earlier
3. If the normalized running sum is greater than the decision boundary for the given country, classify the text as label 0. Otherwise, classify the text as label 1.

Table 4 displays the F1 scores achieved when using only the z-score sum with the optimal decision boundary to classify data. This was all done on perfectly balanced data sets. Figure 2 shows the distribution of the z-score sum for two different languages. As is visible, the mean of the different labels vary in both examples.

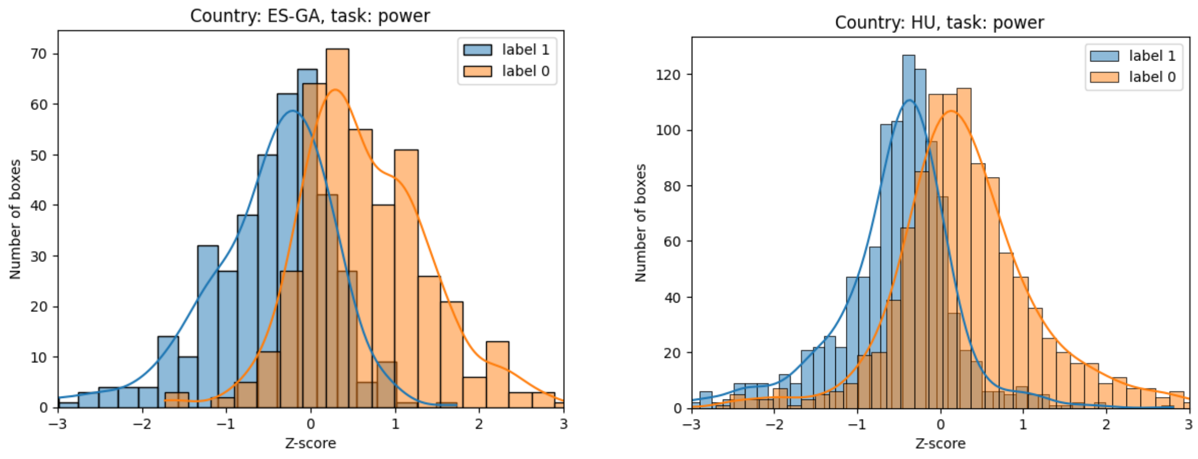

Figure 2: Boxplots of the z-score sum for two different countries.

2.3.3. Combining with BERT

As we have several approaches that perform well on certain texts/parliaments, we wanted to take the best out of the different models. Therefore, for each parliament, we ran the BERT model as well as the model using the discriminatory words. Each BERT prediction is a floating point between 0 and 1. This smoothed result gives us a level of confidence of the model in its prediction. The closer to an extremum a prediction is, the more confident the model is. When using the discriminatory words model, we also

Table 4
Results on the training data when using only the manual feature running Z-score sum with the optimal decision boundary, i.e. classifying all texts with a running Z-score sum less than the decision boundary as 1, and texts with a running Z-score sum greater than the decision boundary as 0. Classification was done on perfectly balanced data sets, meaning that the majority label has been randomly downsampled until a perfect 50/50 split between labels was achieved. Empty cells are parliaments absent from a sub-task.

Country	F1-score on Subtask 1	F1-score on Subtask 2
AT	0.6903	0.7071
BA	0.6960	0.6700
BE	0.7019	0.6757
BG	0.6668	0.7241
CZ	0.6750	0.6690
DK	0.6783	0.7085
EE	0.6971	-
ES	0.6970	0.6994
ES-CT	0.6992	0.7236
ES-GA	0.7712	0.7916
ES-PV	-	0.7335
FR	0.6901	0.6733
GB	0.7222	0.7334
GR	0.6958	0.6926
HR	0.6762	0.6654
HU	0.7355	0.7818
IS	0.7101	-
IT	0.6792	0.7012
LV	0.6834	0.6811
NL	0.6701	0.7108
NO	0.6704	-
PL	0.6888	0.6903
PT	0.6994	0.6802
RS	0.6763	0.7179
SE	0.7133	-
SI	0.6707	0.6835
TR	0.7053	0.7181
UA	0.7286	0.6931

get a confidence value. To aggregate those two results, we decided to select the prediction with the highest confidence.

2.4. SVM

Finally, we trained a Support Vector Machine (SVM). To explain briefly, SVM is a supervised learning algorithm used for classification and regression. It works by finding the hyperplane that best separates data into classes, maximizing the margin between the classes' closest points, called support vectors. SVMs can handle linear and non-linear data using kernel functions to transform the data into higher dimensions.

In our system, we preprocess the texts with a term frequency-inverse document frequency (TF-IDF) vectorizer. Then, we use these vectors in the SVM. The feature used is made of character n-grams, for which we chose to restrain the lower and upper boundaries of the n-grams to one and three respectively. This means that our feature is composed of unigrams, bigrams and trigrams. We decided to test this binary text classification with a fundamental approach, to compare with more complicated and elaborate approaches. In that sense, we did not try to boost this method's performances with a hyperparameter search, but just to have an idea of the power of this approach.

3. Results

In this section, we present our results for our different approaches. In Tables 5, 6 and 7, we use the following abbreviations for the approach name:

- Baseline: Baseline submitted by the organizers of the task
- Bert-basic : Bert trained on the sole parliament's data
- Bert-all-lang: Bert trained on every parliament's data
- Bertaugm: Bert trained with the training's data incremental process
- Llama3: Llama3-70b prompting with zero-shot learning
- Svm: Support Vector Machine
- Logreg: Logistic regression, which is our run on the baseline approach

As we have aforementioned, we have not submitted all methods on all parliaments. Instead, prompting methods were used only on the GB orientation dataset, as well as the Llama 3 fine-tuning method. The logistic regression, SVM, and BERT fine-tuned methods were submitted to every parliament on both sub-tasks (or almost every parliament, some are missing because of mistakes). Finally, the BERT augmented approach has only been submitted on 10 parliaments on the power task: BA, DK, ES, ES-PV, GB, GR, HR, RS, SI, TR, as these were the ones showing a significant increase of the F1-score on our test data.

Table 5
Number of parliaments where a certain approach is the best of all our approaches, both sub-tasks taken together

Approach	Parliaments Count
svm	14
bert-all-lang	11
baseline	10
logreg	8
bert-basic	6
bertaugm	3
Llama3	1

Table 5 exhibits an overview, for each one of our methods, the number of parliaments (on both sub-tasks taken together) where this precise method gave the best F1 score among all methods submitted. For example, out of the 53 parliaments, 14 of them had the best F1-score using SVM as a classifier.

Moreover, Table 6 and Table 7 detail our best submission outcomes for each parliament on the orientation and power sub-tasks, respectively. For each parliament, we can learn our best achieved F1-score along with the approach used to get this result. Additionally, we compare our result with the baseline, and report the improvement over it in terms of F1-score. For example, in Table 7, we can see that our best F1-score for the parliament HU on the power task is 0.89968, obtained using the SVM method. Compared to the baseline (F1-score of 0.857642), we enhanced this score by 3.2326%.

Evaluating the average F1-score on all parliaments of the sub-task 1 (orientation), our best method is BERT trained on all orientation parliaments. The submission of this strategy led to an F1-score of 0.585136, outperforming the baseline by 2.4829% and ranking our team eight out of ten. On the sub-task 2, our best method is the method of BERT with augmented training data, which achieved an F1-score of 0.625254, being 0.14802 behind the baseline, ranking us ten out of eleven.

Even though those results are not showing great improvement on the overall parliaments, some individual results are notable and show promising progress. For example, the SVM approach is also

Table 6
Best Orientation Evaluation per Parliament

Parliament	Our approach	F1 Score	F1 Difference to baseline
TR	baseline	0.840882	+0.0000%
GB	Llama3	0.790132	+4.5201%
ES-GA	svm	0.780211	+11.3434%
SE	baseline	0.749723	+0.0000%
GR	baseline	0.741625	+0.0000%
HU	svm	0.737012	+16.6735%
ES	logreg	0.71786	+0.0365%
ES-CT	svm	0.692944	+3.8107%
PL	bert-all-lang	0.692107	+23.2108%
UA	svm	0.682056	+9.7984%
RS	bert-basic	0.639734	+11.3972%
PT	logreg	0.63335	+0.1516%
NO	baseline	0.615736	+0.0000%
BG	bert-all-lang	0.613049	+7.9860%
BE	svm	0.599305	+14.9923%
AT	bert-all-lang	0.598202	+7.8439%
FR	bert-all-lang	0.57853	+14.9438%
DK	svm	0.570611	+0.6859%
IS	bert-all-lang	0.561041	+17.0720%
HR	bert-all-lang	0.560266	+12.7540%
IT	bert-all-lang	0.558772	+0.1902%
NL	bert-all-lang	0.55518	+4.2632%
FI	svm	0.551095	+0.8286%
LV	bert-all-lang	0.535344	+8.1636%
SI	svm	0.532712	+14.1417%
BA	bert-all-lang	0.526129	+11.0575%
EE	bert-all-lang	0.525498	+5.0717%
CZ	baseline	0.518866	+0.0000%

good for some parliaments. Overall, the average F1 score decreased compared to the baseline (around 1.5% for orientation and 3.5% for power), but it is important to note that it allowed us to increase the F1 score a lot on some parliaments, up to 17.6953% on the LV power parliament, as can be seen in Table 7. The SVM method achieved great results on some parliaments of sub-task 2 (power): it led us to an F1 score of 0.889968 for the HU parliament, setting our team at the second place (out of ten) of the ranking; and achieved F1 scores of 0.880689 for ES-GA and 0.846702 for ES-CT, both setting us at the third place for these two parliaments. It is the same for sub-task 1 (orientation), where we achieved 0.692944 with the SVM, setting us to the second place (out of eleven).

Fine-tuning BERT was found to be a beneficial approach for certain parliaments. The training of BERT using all available parliament data resulted in our team achieving the second position in the sub-task 1 for the BA parliament, achieving an F1 score of 0.526129.

Significant results were also observed for a specific parliament in sub-task 1 (orientation). Prompting Llama 3 demonstrated effectiveness, particularly for the GB orientation parliament, achieving an F1 score of 0.790132. This approach secured the second position out of ten methods in the final ranking for this parliament.

Table 7
Best Power Evaluation per Parliament

Parliament	Our approach	F1 Score	F1 Difference to baseline
HU	svm	0.889968	+3.2326%
ES-GA	svm	0.880689	+5.1572%
ES-CT	svm	0.846702	+6.7065%
TR	logreg	0.836893	+0.5902%
PL	logreg	0.767386	+0.8908%
ES-PV	svm	0.742512	+2.7673%
RS	bertaugm	0.741145	+9.3724%
GB	logreg	0.722565	+1.1512%
LV	svm	0.688353	+17.6953%
BG	baseline	0.681117	+0.0000%
HR	bert-basic	0.68003	+7.7972%
AT	logreg	0.666726	+0.6524%
FR	bert-basic	0.666391	+0.7619%
GR	bert-basic	0.664713	+3.7555%
CZ	logreg	0.654461	+1.8956%
ES	baseline	0.652175	+0.0000%
IT	bert-basic	0.639566	+21.3734%
NL	logreg	0.636119	+2.0396%
DK	bertaugm	0.629958	+6.8779%
PT	baseline	0.619798	+0.0000%
BE	baseline	0.612543	+0.0000%
SI	bertaugm	0.602376	+6.9857%
BA	svm	0.562684	+10.8886%
FI	baseline	0.561368	+0.0000%
UA	bert-basic	0.533312	+7.2237%

4. Conclusion

In this paper, we described our different approaches for the ideology and power identification in parliamentary debates shared task, proposed by the Touché Lab at CLEF 2024.

We have reported several approaches, including fine-tuning different LLMs, prompting Llama 3, as well as examining a classical SVM which proved to be worth of interest. We discovered that one approach is not always working the best for every parliament, but can show interesting results for some of them. Because of this, we could achieve better results by creating a rule based approach, and using the method that individually works best for each parliament. We plan in our future work to integrate an argument mining phase in the classification pipeline, since it proved to be essential in different applications like comparative question answering [11], and financial speech analysis [12]. Thus, we can use argumentation to analyze the given debate speech.

We also plan to extend on the prompting approach to non-English-speaking texts. Indeed, Llama 3 currently works with English-speaking data, but as Meta is working on training Llama 3 on multilingual data in a near future, it could become an effective solution for other languages soon. In addition, examining GPT capabilities on political speech analysis would be interesting.

References

[1] J. Kiesel, Ç. Çöltekin, M. Heinrich, M. Fröbe, M. Alshomary, B. D. Longueville, T. Erjavec, N. Handke, M. Kopp, N. Ljubešić, K. Meden, N. Mirzakhmedova, V. Morkevičius, T. Reitis-Munstermann, M. Scharfbillig, N. Stefanovitch, H. Wachsmuth, M. Potthast, B. Stein, Overview of Touché 2024: Argumentation Systems, in: L. Goeuriot, P. Mulhem, G. Quénot, D. Schwab, L. Soulier, G. M. D. Nunzio, P. Galuščáková, A. G. S. de Herrera, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Fifteenth International Conference of the CLEF Association (CLEF 2024), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2024.

[2] A. Alhamzeh, E. Egyed-Zsigmond, D. E. Mekki, A. E. Khayari, J. Mitrović, L. Brunie, H. Kosch, Empirical Study of the Model Generalization for Argument Mining in Cross-Domain and Cross-Topic Settings, in: Transactions on Large-Scale Data-and Knowledge-Centered Systems LII, Springer, 2022, pp. 103–126.

[3] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, CoRR abs/1810.04805 (2018). URL: http://arxiv.org/abs/1810.04805. arXiv:1810.04805.

[4] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, I. Androutsopoulos, LEGAL-BERT: The muppets straight out of law school, in: Findings of the Association for Computational Linguistics: EMNLP 2020, Association for Computational Linguistics, Online, 2020, pp. 2898–2904. doi:10.18653/v1/2020.findings-emnlp.261.

[5] L. Zheng, N. Guha, B. R. Anderson, P. Henderson, D. E. Ho, When Does Pretraining Help? Assessing Self-Supervised Learning for Law and the CaseHOLD Dataset, in: Proceedings of the 18th International Conference on Artificial Intelligence and Law, Association for Computing Machinery, 2021. arXiv:2104.08671.

[6] P. Henderson, M. S. Krass, L. Zheng, N. Guha, C. D. Manning, D. Jurafsky, D. E. Ho, Pile of Law: Learning Responsible Data Filtering from the Law and a 256GB Open-Source Legal Dataset, 2022. URL: https://arxiv.org/abs/2207.00220.

[7] H. Yadav, Dropout in Neural Networks, 2022. URL: https://towardsdatascience.com/dropout-in-neural-networks-47a162d621d9.

[8] S. Vishwakarma, How to Understand Sigmoid Function in Artificial Neural Networks?, 2023. URL: https://www.analyticsvidhya.com/blog/2023/01/why-is-sigmoid-function-important-in-artificial-neural-networks/.

[9] AI@Meta, Llama 3 model card (2024). URL: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

[10] J. Savoy, Trump's and Clinton's Style and Rhetoric during the 2016 Presidential Election, Journal of Quantitative Linguistics 25 (2018) 168–189. URL: https://doi.org/10.1080/09296174.2017.1349358. doi:10.1080/09296174.2017.1349358.

[11] A. Alhamzeh, M. Bouhaouel, E. Egyed-Zsigmond, J. Mitrović, L. Brunie, H. Kosch, Query Expansion, Argument Mining and Document Scoring for an Efficient Question Answering System, in: A. Barrón-Cedeño, G. Da San Martino, M. Degli Esposti, F. Sebastiani, C. Macdonald, G. Pasi, A. Hanbury, M. Potthast, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction, Springer International Publishing, Cham, 2022, pp. 162–174.

[12] A. Alhamzeh, Language Reasoning by means of Argument Mining and Argument Quality, Ph.D. thesis, Universität Passau, 2023.