I've Seen Things You Machines Wouldn't Believe: Measuring Content Predictability to Identify Automatically-Generated Text

Piotr Przybyła[1,2,*], Nicolau Duran-Silva[1,3] and Santiago Egea-Gómez[1]

[1] *LaSTUS Lab, TALN Group, Universitat Pompeu Fabra, Barcelona, Spain*
[2] *Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland*
[3] *SIRIS Lab, Research Division of SIRIS Academic, Barcelona, Spain*

Abstract

Modern large language models (LLMs), such as GPT-4 or ChatGPT, are capable of producing fluent text in natural languages, making their output hard to manually differentiate from human-written text. However, there are many real-world scenarios where this distinction needs to be made, raising the need for automatic solutions. Here we present our approach to the problem, implemented with the 'AuTexTification: Automated Text Identification' shared task. The core of our model is aimed at measuring 'predictability', i.e. how likely given text is according to several LLMs. This information, supplemented with features describing grammatical correctness, word frequency, linguistic patterns and combined with fine-tuned LLM representation is used to train a neural network on the provided datasets. The resulting model achieves the best performance among the submissions in subtask 1 (differentiating between human- and machine-generated text), both for English and Spanish. We also provide the results of our internal topic-based evaluation to show strengths and weaknesses of different variants of our contribution.

Keywords

machine-generated text, large language models, predictability, automatic text identification

1. Introduction

Modern large language models (LLMs) can generate increasingly natural-sounding text: remarkably fluent, grammatically correct and difficult to distinguish from human-written content [1, 2, 3]. Especially the most recent models, including GPT-4, BLOOM, PaLM and ChatGPT, produce more coherent and creative text than previously [4], demonstrate astonishing capabilities in writing realistic short and long documents [5] and are able to answer complex questions involved in various types of exams intended for humans [6]. Studies have shown that while human participants are able to differentiate the real text and GPT-2 generations, they fail in case of GPT-3 [7].

IberLEF 2023, September 2023, Jaén, Spain

*Corresponding author.

piotr.przybyla@upf.edu (P. Przybyła); nicolau.duransilva@sirisacademic.com (N. Duran-Silva); santiago.egea@upf.edu (S. Egea-Gómez)
0000-0001-9043-6817 (P. Przybyła); 0000-0001-5170-4129 (N. Duran-Silva); 0000-0002-4359-0202 (S. Egea-Gómez)

The dangers brought by proliferation of automatically-generated content have been recognised in the Natural Language Processing (NLP) community for some time [8], but they have become a topic of wider discussion following the release of ChatGPT. Some of the most concerning applications include generating fake news [9], impersonating real people in social media by bots [10], preparing massive numbers of fake reviews [11] or disrupting text-based evaluation in education [12]. However, much of the negative impact in these domains could be avoided with high-accuracy tools for detecting automatically-generated text.

Due to this motivation, the problem of detecting automatically-generated text has been attracting attention in recent years (see section 3). Here we present our solution to the problem, prepared in the framework of the shared task *AuTexTification: Automated Text Identification* [13], a part of *5th Workshop on Iberian Languages Evaluation Forum (IberLEF 2023)* [14].

At the core of our solution is the notion of *predictability*, expressing how likely or believable a given text is according to a language model. We assume that an LLM will assign higher probability to automatically-generated text, especially when the generator and detector have similar architecture. The text written by humans, on the other hand, is likely to be surprising or less likely to LLMs. This intuition is expressed as features and supplemented with measures of grammatical correctness, word frequency and language patterns. The final solution, trained on the data made available within the shared task, achieves the best performance from all submitted solutions in subtask 1 (differentiating human- and automatically-generated text), both in English and Spanish. We also provide classifiers for subtask 2, aimed at predicting a particular model that was used to generate a given document. The code for the best of our solutions is openly available (https://github.com/piotrmp/autext).

2. Task

The *AuTexTification: Automated Text Identification* shared task [13] aims to address the detection of content created by text generation models in English and Spanish. Several different text genres are included, which include legal or administrative documents, how-to articles and social media.

The shared task is divided in two subtasks:

- **Subtask 1 - Human or Generated**: a binary classification problem where participants have to build systems able to distinguish between machine- or human-generated text. For each language, Spanish and English, a balanced dataset is provided.
- **Subtask 2 - Model Attribution**: a multi-class classification problem where participants build systems able to attribute authorship to 6 generative models, with sizes ranging from 2B to 175B. Authorship is labelled with six classes (A, B, C, D, E, and F), but participants do not know, which generated model is behind the classes. For each language, Spanish and English, a balanced dataset is provided.

Table 1 includes examples from different genres, comparing text generated by machine and humans for both English and Spanish. Table 2 shows corpus statistics, including data size and text length. For subtask 1, the size of test size is relatively large compared to training data, which makes the evaluation more challenging. In subtask 2, test represents a 25% of training

Machine	Human
English – legal text	
Article 1 This Regulation lays down detailed rules for the implementation of Council Regulation (EC) No. 1234/2007 as regards the import of milk and milk products from third countries. Article 2 1. Imports of milk and milk products from third countries shall be subject to the following conditions: (a) The product shall be accompanied by a health certificate issued by the competent authority of the exporting country and by a certificate of origin; (b	*Article 4a (c) of Regulation (EEC) No 1336/86 is hereby replaced by the following: (c) except for applications submitted before 31 January 1991, Germany shall be authorized, by way of derogation from the first subparagraph of Article 1 (1), to grant such compensation for an equal reduction in the relevant part of the production of each party concerned in relation to previous production. Article 2 This Regulation shall enter into force on the day of its publication in the Official*
English – social media	
@swfacts LOL! Thats too funny. @swfacts We know! We cant help but laugh.	*@drellbee when are we going lens shopping? I want a macro now i will research*
English – how-to	
Cut off the ends of the carrots and discard them. Peel the carrots with a vegetable peeler or a sharp knife. Slice or chop the carrots as desired.	*Add the egg yolks and cornstarch to the same bowl and whisk well. Continue whisking until the mixture seems smooth and the contents are evenly distributed. Pour the remaining milk in a medium saucepan. Add the lemon zest,*
Spanish – legal text	
Artículo 1 Modificación de los importes aplicables Los importes aplicables a los productos agrícolas objeto de la aplicación de los Reglamentos (CE) n°s 1442/95, 1443/95, 1444/95, 1445/95, 1446/95 y 1447/95, según se establece en los anexos I, II, III, IV, V y VI de los citados Reglamentos, se modifican de conformidad con los importes que figuran en el anexo I del presente Reg	*Artículo 1. Se modifica el Anexo del Reglamento (CEE) n º 3800/81 de acuerdo con las indicaciones comprendidas en el Anexo del presente Reglamento. Artículo 2. El presente Reglamento entrará en vigor el día de su publicación en el Diario Oficial de las Comunidades Europeas. El presente Reglamento será obligatorio en todos sus elementos y directamente aplicable en cada Estado miembro. Hecho en Bruselas, el 19 de diciembre de 1983.*
Spanish – social media	
@ElDatoDelDia: Un pescador chino se ha convertido en millonario tras vender un pez de 1,8 millones de yuanes o US$230	*@SantiContreras: Las fotos de tu iPhone podrían ser tu nueva contraseña para todo: http://t.co/0nxCAkeMUr*
Spanish – how-to	
Cuando el agua esté hirviendo, agrega la pasta de tomate y cocina durante 5 minutos o hasta que espese. Si es necesario, puedes agregar un poco más de agua a la sartén para evitar que se queme. Coloca una rebanada de masa en una placa para hornear ligeramente engrasada. Deja enfriar por unos 30 segundos antes de cortarlos en cubitos grandes. Una vez cortados, coloca los tomates en una fuente apta para horno. Calienta aceite	*Córtala en dos para formar las masas de dos pizzas pequeñas. Así será más fácil manipularlas y entrarán en un horno casero. Trabaja las bolas por separado cuando sigas las instrucciones que se presentan a*

Table 1
Examples of machine and human generated texts in training data of AuTexTification shared task.

data size. Class distribution in training datasets is balanced, and the average number in each example is around 60, with a lower number in training data for subtask 1, probably because of the contribution of social media samples, which are shorter.

3. Related work

Automatic analysis of AI-generated content has mainly revolved around the tasks of (1) binary classification of human- and machine-generated text, which is called *machine vs. human* problem [3, 5]; and the (2) multi-class problem of authorship attribution for different generative models [5, 15].

The methods proposed for detecting machine-generated text can be split into two main types. On the one hand, feature-based [3] or metric-based approaches [16] consist of creating feature

	Train data		Test data	
Subtask	#Instances	Avg. length	#Instances	Avg. length
Substask 1 - EN	33,845	53.7	21,832	62.6
Substask 1 - ES	32,062	52.1	20,129	62.8
Substask 2 - EN	22,416	60.0	5,605	59.8
Substask 2 - ES	21,935	57.8	5,482	57.9

Table 2
Dataset description for the two subtasks in English and Spanish.

vectors from input sequences, then used to train a classification model. On the other hand, model-based approaches [3, 16] leverage the potential of pre-trained language models. There have also been proposed hybrid approaches which combine strengths of both strategies [17].

Feature-based approaches are more computationally efficient, but less transferable between different architectures [3, 18]. The generated features are based on word frequency, output of LLMs, fluency or reliability, or linguistic patterns detected by auxiliary models [3]. Among the token-wise contextual features leveraging LLMs are: log-probability, rank or entropy of words [16]. Approaches used for classification include SVMs, random forests, or neural networks [3].

Model-based approaches using Transformer LLMs are more extensively used in recent research, and they seem to perform better than feature-based solutions [16, 15]. The fine-tuning of a RoBERTa model [19] on a dataset of human- and machine-generated text is a particularly popular strategy in literature[20, 5, 15, 17], which outperforms other classifiers in different domains, like biomedical and technical documents [20] or tweets [17]. There have also been more elaborate approaches, including concatenation of outputs from a fine-tuned and unchanged RoBERTa model, fed to a feed-forward neural network [17]. Other recent methods rely on the use of dependency trees for identifying factual inconsistencies in a conversational setting [21].

Fewer researchers have focused on the authorship attribution variant [22, 23, 5, 15]. Some have shown that n-grams and POS-tags are not enough for solving the problem, requiring word embeddings [22]. Others have noticed that random forests based on linguistic features obtain the highest score, beating models combining word embeddings, CNN and LSTM, and comparable to fine-tuned RoBERTa [5]. However, in a more recent research, a fine-tuned RoBERTa outperforms feature-based methods [15].

Training and evaluation data collection is key, because the class distribution [3] and the genre distribution [2] of the dataset used should resemble real-world applications. In the case of neural-network-based methods, there is a tendency to overfit to topics present in the data a detection system was trained on [24]. Most research has focused on English texts, but as the multilingual capacity of LLMs increases, the challenge of addressing other languages arises as well [3].

Significant effort has also been put in understanding how humans can differentiate between machine- and human-generated texts, focusing on generators' errors that can be used [1, 2], differences in task difficulty between genres [2], and versions of LLM used for generation [7]. It appears that humans are increasingly likely to be fooled by more powerful generative models, but automated detection methods can have more opportunities to improve [3]. However, prediction models that cover aspects relevant to humans could be the best approach [25].

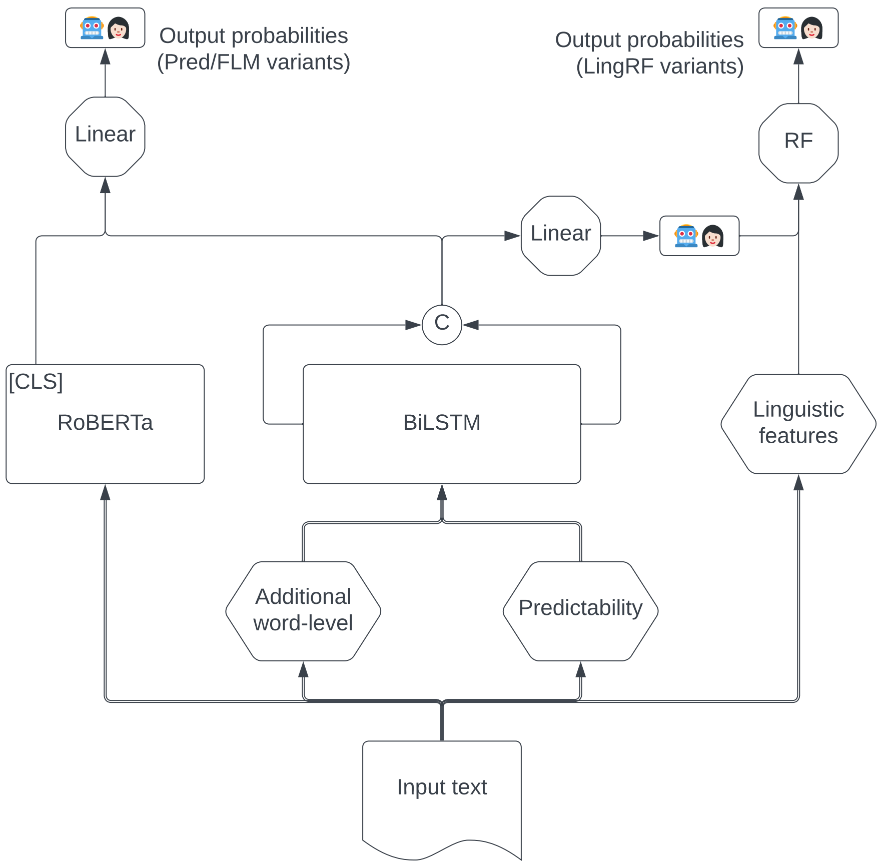

Figure 1: An architectural template used to build different variants of classifiers for detecting machine-generated text.

4. Methods

In this section, we describe the design and implementation of our automatic text identification solution – see figure 1 for a broad architectural overview. The basic building blocks are described in section 4.1, while section 4.2 enumerates the classifier variants that are evaluated within this work. Section 4.3 outlines the training process and implementation details.

The common core of most variants is a set of features measuring *predictability*, expressing how likely the given input text is according to several generative language models (section 4.1.1). These might by supplemented with additional features, checking the grammatical correctness and frequency of words (section 4.1.2). All token-level features are fed to a bidirectional LSTM network [26] with a hidden size of 64 to convert these sequences into representation of fixed length of 128 (concatenated from both directions).

The LSTM representation can be concatenated with an output from a fine-tuned language model (section 4.1.3) and transformed by a dense linear layer, followed by softmax, to return output probabilities. Alternatively, some text-level linguistic features could be collected directly from text (section 4.1.4). To combine them with predictability measurement, we train a small network based on BiLSTM, followed by dense linear layer and softmax to output probabilities. These can then be added to the linguistic features and processed by a random forest classifier [27], returning final class probabilities.

Note that the design remain exactly the same for both languages and subtasks, differing only in dimensionality of some data elements due to various language models used and required number of output classes.

4.1. Building blocks

4.1.1. Predictability measurement

The goal of the core features of our approach is to express how likely a given sequence of tokens is according to a generative language model. We expect machine-generated content to be less surprising and more predictable than text from human authors, especially if the models used for generation and predictability are similar or the same. This strategy has previously been applied to differentiating between social media bots and real users [28]. The fact that language models assign high probability to repetitive, low-diversity generated sequences and lower to text from human authors is a major weakness of the popular sampling strategies [29].

For a given token sequence $t_{0..n-1}$ of length n, we use a generative language model glm to obtain the probability of producing a token w^j (from dictionary D) at i-th position:

$$p(i, w^j) = P_{glm}(t_i = w^j | t_{0..i-1})$$

We use this quantity to obtain the following three types of features:

- Log-probability of the observed token t_i^*, measuring how likely it is given the model:
$$lt_i = \log p(i, t_i^*)$$
- Log-probability of the most likely token according to the model, measuring the confidence of the model's top prediction:
$$lm_i = \max_{j \in D} \log p(i, w^j)$$
- Entropy of the token probability distribution at a given position, measuring the uncertainty of choosing the next token according to the model:
$$H_i = -\sum_{j \in D} p(i, w^j) \log p(i, w^j)$$

These features are computed for each token position, except the first one, for a length up to 128. Additionally, a binary mask feature is added to account for different lengths of the input text.

Notice that the generative model used to compute predictability, denoted by glm, does not have to be identical to the one used to generate the considered text. However, their similarity is helpful in improving performance and can be expected, since virtually all of the LLMs are based on the same architecture, namely Transformer. We use the following generative language models of different sizes:

- For English: DistilGPT2 [30], GPT-2, GPT-2 Medium and GPT-2 Large [31].
- For Spanish: GPT2-base-bne and GPT2-large-bne [32].

As a result, each token is represented by $3 \times 4 + 1$ (for English) or $3 \times 2 + 1$ (for Spanish) predictability features.

Note that the negated average of lt_i over the whole text is equivalent to perplexity. However, instead of using averaging or similar aggregation function, we choose to keep the whole sequence of features and let a recurrent neural network learn to find patterns resulting from the sampling strategy.

4.1.2. Additional word-level features

We employ two types of additional word-level features: word frequency and grammatical correctness.

Frequency features can play a role, since the distribution of token frequency in machine-generated text has been shown to not correspond to that of human-written text [3]. Moreover, from a wide range of errors produced by generative models that humans recognise [1], such as related to common sense, grammar, fluency, redundancy or self-contradiction, we choose to recognise grammar errors.

Word frequency is obtained from the version 3 of Google Book Corpus Ngrams dataset [33] (obtained from https://storage.googleapis.com/books/ngrams/books/datasetsv3.html), which is available for English and Spanish, among many other languages. Google Books includes over 200 billion words in 40 million documents, providing a good approximation for likelihood of encountering a word in print, even a very rare one. We assume frequency of 1 for those words that do not appear in the dataset and apply logarithm to the number of occurrences. To coordinate the frequencies with the predictability features described above, the tokenisation used to distinguish words (space-based) needs to be aligned with the tokenisation used by language models (wordpiece-based). In the typical case of a single long word corresponding to multiple subword tokens, we assign the frequency of the full word to all its tokens. However, more complicated situations are also possible, all of which are handle in a way guaranteeing the same length of feature vector coming from both sources.

We apply grammar check by open-source LanguageTool (https://github.com/languagetool-org/languagetool), using the Python wrapper (https://github.com/jxmorris12/language_tool_python). The tool is available for both Spanish and English as API service, which we use through a caching mechanism to avoid repeated queries. The system returns a sentence without grammar errors, which we compare to the original, assigning value 1 to the preserved words and 0 when the tool made modifications to fix errors. The resulting vector is aligned to the LLM tokenisation, as with frequency features.

4.1.3. Fine-tuned language model

Given the limited size of the training data, we hope to improve performance in the transfer learning schema by including a text representation from a fine-tuned large language model. We employ the RoBERTa [19] models: the Base variant for English and Base BNE [19] for Spanish.

The pooler output based on `[CLS]` token representation of length 768 is used, and the model is fine-tuned with the whole network (see 4.3 for details).

4.1.4. Text-level linguistic features

Another line we explored is using linguistic features, like Part-Of-Speech (POS) or Word-Dependency (WD) labels, to solve both sub-tasks. These features are aggregated at the level of the whole document, providing a more high-level view of the text.

We expect automatic text generation models might imperfectly mimic the natural human writing, i.e. repeat words and punctuation, produce grammatical and morphological errors uncommon for humans and alter or 'invent' words. These situations can by captured thanks to the available linguistic resources for English and Spanish. For example, annotating morphological categories can give information about errors in producing words from subwords. The subword tokenisation used in LLMs can generate words sharing the root of an actual word, but altering its meaning when an incoherent prefixes and suffixes are erroneously concatenated. Additionally, altering suffixes and prefixes in a word might also change its linguistic function in the whole sentence.

With this motivation, we annotated every sentence with morphological, word-dependency, part-of-speech and name-entity features. We also incorporated word frequency indicators (using Google Books dataset as described in section 4.1.2) to identify words that are so uncommon they might have been 'invented' by text generation models. As a result, we generated per-word sequences of linguistic categories representing all those features.

Finally, we trained a Random Forest (RF) classifier [27] to exploit the linguistic information and augment the performances of our system. As it is not feasible to input sequences in RF, we processed the linguistic sequences to compute statistics on each linguistic category, for instance the percentage of adverbs and the percentage of zero-frequency words appearing in a sentence.

4.2. Classifiers

Based on the overall architecture outlined above, we selected the following variants for evaluation:

- **Pred**: only the LSTM network using predictability features,
- **FLM**: only the fine-tuned language model (baseline),
- **Pred+FLM**: a linear classifier using the representation from fine-tuned language model and LSTM (submitted to shared task as `Hybrid`),
- **Pred+FLM+Add**: as above, but including also the additional token-level features (submitted to shared task as `Hybrid+`),
- **LingRF**: a random forest using linguistic features.
- **LingRF+PredOut**: a random forest using linguistic features and probabilities returned by an LSTM operating on the predictability features (submitted to shared task as `LinguisticRF`).

4.3. Implementation and training process

The neural network component is implemented in *PyTorch* [34], while the language models are loaded through *Huggingface Transformers* [35]. We use the Adam optimiser [36] with a learning rate of 10^{-3} (for training BiLSTM) and 2×10^{-5} (for fine-tuning a language model). In case of variants combining both components, we use the larger learning rate while keeping the weights of the language model frozen for the first 5 epochs, and then continue training the whole network with the smaller learning rate.

In topic-based experiments (see section 5) we always train for 10 epochs. However, for preparing the final submission, we randomly split the available data into 20% development subset and the training subset. The model is trained for 20 epochs, and the F-score on the held-out dataset is used to choose the best configuration. To further reduce overfitting, we implement early stopping by choosing the first epoch, when at least an F-score of $(f_{max} - 0.01)$ is reached, where f_{max} denotes the best observed performance on the development set. The test predictions made by a model selected in that way are submitted to the shared task.

Regarding the linguistic RF models, we used `scikit-learn` [37] and applied model selection to tune the number of trees and their depth. As variants using 200 trees and the maximum tree depth of 60 obtained the best or close-to-the-best scores, we decided to choose this setting for all subtasks and languages. We use the models implemented in `spaCy` [38] library for annotating the linguistic features. Namely, we employ `en_core_web_sm` and `es_core_news_sm` to tag the English and Spanish datasets, respectively. Afterwards, we aggregate the different categories in counts. In the case of word-frequency attributes, we use the Google Books N-grams to count the number of words not appearing in the resource and the number of words having very high and medium-low frequencies. For English, we obtain 165 and 163 predictors in the subtask 1 and 2, respectively. For Spanish, we generate 299 predictors for subtask 1 and 288 for subtask 2.

5. Evaluation

In an initial experimental stage, we fit a RF with default parameters to verify the importance of linguistic features for the English subtask 1. Interestingly, the top 5 features with the highest importance scores are: VERY_FREQ_W (number of very frequent words), NO_FREQ_W (number of words not appearing in the unigram list), MEDIUM_LOW_FREQ_W (number of words with an average frequency), empty NER label (no entity) and DATE (NER category to represent dates).

Our main evaluation has two components: internal and external. The training-test split and data collection procedures for external evaluation are provided by the shared task organisers [13].

The purpose of the internal evaluation is to assess the performance of different variants of our solution and understand which one performs the best on previously unseen data. The challenge here is that the properties of automatically-generated text will strongly depend on the prompt used to generate it, which remains hidden from us. Performing evaluation on a random split of existing data may favour solutions that overfit to the particular types or domains of observed text and underperform on truly new and out-of-domain data.

Thus, we have decided to conduct evaluation in cross-validation over topically-split data.

	Subtask 1				**Subtask 2**			
	English		**Spanish**		**English**		**Spanish**	
Variant	**F1**	**SD**	**F1**	**SD**	**F1**	**SD**	**F1**	**SD**
FLM (baseline)	0.84	0.0598	0.87	0.0762	0.52	0.0399	**0.57**	**0.0168**
Pred	0.85	0.0269	0.81	0.0637	0.43	0.0552	0.45	0.0180
Pred+FLM	**0.93**	**0.0114**	0.90	0.0495	**0.54**	0.0275	0.56	0.0288
Pred+FLM+Add	**0.93**	0.0221	**0.91**	**0.0170**	**0.54**	0.0262	0.56	0.0227
LingRF	0.74	0.0440	0.64	0.1706	0.38	**0.0177**	0.40	0.0173
LingRF+PredOut	0.89	0.0122	0.81	0.0900	0.47	0.0404	0.49	0.0217

Table 3
Internal evaluation: the overall F1-score and its standard deviation (SD) over CV folds achieved by the tested variants of our solution in both subtasks and languages. The best value in each column (highest F1, lowest SD) is highlighted by boldface.

This fact implies that each model is evaluated on data from different topics (genres, types, domains…) than the sample seen by the models during the training, which is likely to resemble the real-world usecases.

This procedure is carried out by fitting Latent Dirichlet Allocation topic modelling implemented in `scikit-learn` [37] to detect 10 document topics. Therefore, each document is linked to its most likely topic and the 10 topics are paired into 5 folds for cross-validation. For merging topics, we order topics by number of documents and then merged each topic with its inverse position in the list in order to create folds of similar size.

The performance metrics used in the evaluation stages is macro-averaged F1-score. As the data are relatively balanced with respect to class labels (see details in section 2), we do not use other metrics, more sensitive to unbalanced data distributions (such as AUC-ROC or G-mean).

6. Results

Table 3 shows the results of our internal cross-validation experiments. We use the different topic folds to report the overall F1 score (calculated using aggregated predictions from all folds) and the standard deviations (of F1 results from different folds) to quantify the performance variations due to topic diversity.

We can see that the Pred+FLM+Add achieves overall best results, but the performance of Pred+FLM is within 1% margin, making it hard to assess the contribution of the grammar- and frequency-based features. Subtask 2 in Spanish is the only configuration, where our solutions appear unable to beat the baseline.

Table 4 shows the external evaluation results, including the top 10 submissions for each subtask-language combination. The Pred+FLM+Add obtains the best performance in subtask 1, in line with internal evaluation. However, it is interesting to notice that the improvement made thanks to the additional features is much larger here, reaching 0.05 in F1-score.

	Subtask 1				**Subtask 2**			
	English		**Spanish**		**English**		**Spanish**	
No	**Name**	**F1**	**Name**	**F1**	**Name**	**F1**	**Name**	**F1**
1	**Pred+FLM+Add**	0.81	**Pred+FLM+Add**	0.71	Drocks-3	0.62	Drocks-2	0.65
2	**Pred+FLM**	0.74	Linguistica_F-P_et_al-1	0.71	Drocks-1	0.61	Drocks-3	0.65
3	CIC-IPN-CsCog-2	0.74	RoBERTa-baseline	0.69	Drocks-2	0.61	Drocks-1	0.64
4	LastMinute-2	0.74	**Pred+FLM**	0.68	ViDA-1	0.61	**Pred+FLM**	0.62
5	Drocks-2	0.73	KInIT_team_Macko-1	0.68	DeBERTa-baseline	0.60	ELiRF-VRAIN-2	0.62
6	GPLSI_TEAM-1	0.73	andreipreda-3	0.67	BERT4Ever-3	0.60	ELiRF-VRAIN-3	0.62
7	Drocks-1	0.72	andreipreda-2	0.67	**LingRF+PredOut**	0.60	**Pred+FLM+Add**	0.61
8	CIC-IPN-CsCog-3	0.72	GPLSI_TEAM-2	0.67	BERT4Ever-1	0.60	RoBERTa-baseline	0.61
9	blade-runner-3	0.71	turing_testers-2	0.66	BERT4Ever-2	0.59	ELiRF-VRAIN-1	0.61
10	GPLSI_TEAM-3	0.71	PFMP_PoliMi-1	0.66	OD-21-1	0.58	**LingRF+PredOut**	0.61
	(66 more)		(42 more)		(28 more)		(19 more)	

Table 4
External evaluation: the top 10 submissions evaluated in the shared task with the F1-scores. The solutions described here are highlighted, for information on the others see the overview publication [13].

7. Discussion

The results indicate the predictability-based methods perform quite well, with Pred+FLM+Add achieving the best performance in subtask 1 among all submitted approaches. However, it is interesting to notice that none of the simpler approaches reaches the same performance in the external evaluation, showing all three of the feature types included (predictability, LLM representation, additional features) play an important role. This indicates that detecting machine-generated text is a multifaceted problem and requires taking into account different properties of the content. In particular, the additional token-level features delivered marginal gains in internal evaluation, but yielded large improvement in the external one, suggesting they could be helpful in avoiding overfitting to data available during training.

Regarding LingRF, we observe that it performs relatively well in subtask 1 in spite of its simplicity. This fact reveals that mining linguistic feature can uncover informative patterns to classify human/machine texts. On subtask 2, the F1-score dropped notably due to the higher task complexity. In the case of LingRF+PredOut, it is apparent how combining linguistic information with predictability injects extra knowledge to the classifier. For all tasks, except for Spanish subtask 1, the F1 improved comparing to using only Pred or LingRF. Observing 4, we see that LingRF+PredOut is in the top-10 algorithms for subtask-2 obtaining a F1 only 2 and 4 points lesser than the winner for English and Spanish, respectively. In the instance of subtask-1, it obtained the position 12 with a F1 of 70 for English and 64.7 for Spanish.

Despite promising results, our work has several limitations. Firstly, our solution combines numerous feature types defined at different levels: whole document, words or wordpiece tokens. While we have managed to explore some variants of combining them, we believe further performance gains are possible through deeper integration, e.g. by linking predictability and linguistic information and processing the resulting sequence through a recurrent neural network or a Transformer.

Secondly, we have only explored the simplest training-test framework, imposed by the shared task schema. However, the automatic text identification differs from other text classification problem in that it is very easy to obtain additional labelled data – by using extensive corpora with human-written content, as well as generating additional text from LLMs. This makes possible evaluating more sophisticated frameworks, including contrastive learning or adversarial training.

Thirdly, our evaluation was necessarily limited by the fact we do not know the details of the text generation techniques used to prepare the shared task input, especially the LLM generators and prompts that were employed. Having access to this information would allow answering questions important for practical applications, such as (1) how the detection accuracy depends on the generator strength, (2) whether using the same LLM for preparing the generator and detector is a prerequisite to good performance or (3) how well our solution performs on text from a domain unseen in training. We have also been able to evaluate our approach only on two languages, showing very different results.

Finally, our methods rely on surface-level features and do not attempt to involve any kind of deep understanding of the text meaning. However, the inspection of machine-generated content reveals factual and common-sense errors that could be used as well. For example, in table 1 we can notice a redundant repetition of a user name (second row), incorrect conversion between currencies (fifth row) and inconsistent culinary procedure (sixth row). We hope that more sophisticated methods for detecting machine-generated text could leverage such errors.

8. Conclusions

In this paper, we described the solutions we explored for automatic detection of machine-generated text and authorship attribution to generative models, proposed in the "AuTexTification: Automated Text Identification" shared task. Our system combines feature and model-based approaches. Specifically, we propose a method for measuring 'predictability' of a text sequence according to several LLMs, combined with complementary measures of grammatical correctness, word frequency, and language patterns. The evaluation performed through the shared task shows good performance of our solutions.

We hope that both our contribution and the shared task in general will encourage further research on this area. A lot of exploration is needed before such methods become useful in practice, including the performance of detection on other languages, sensitivity to unseen genres and generators or usefulness of more sophisticated and robust training frameworks.

Acknowledgments

This work is part of the ERINIA project, which has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101060930. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

This work has also been funded by MCIN/AEI /10.13039/501100011033 under the Maria de Maeztu Units of Excellence Programme (CEX2021-001195-M) and supported by the Industrial Doctorates Plan of the Department of Research and Universities of the Generalitat de Catalunya.

References

[1] Y. Dou, M. Forbes, R. Koncel-Kedziorski, N. A. Smith, Y. Choi, Is gpt-3 text indistinguishable from human text? scarecrow: A framework for scrutinizing machine text, in: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2022, pp. 7250–7274.

[2] L. Dugan, D. Ippolito, A. Kirubarajan, S. Shi, C. Callison-Burch, P. Zhou, A. Zhu, J. Hu, J. Pujara, X. Ren, et al., Real or fake text? investigating human ability to detect boundaries between human-written and machine-generated text, in: The 37th AAAI Conference on Artificial Intelligence (AAAI 2023), arXiv, 2023, p. 104979.

[3] E. Crothers, N. Japkowicz, H. Viktor, Machine generated text: A comprehensive survey of threat models and detection methods, 2023. arXiv:2210.07321.

[4] K. Ethayarajh, D. Jurafsky, The authenticity gap in human evaluation, in: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022, pp. 6056–6070. URL: https://aclanthology.org/2022.emnlp-main.406.

[5] A. Uchendu, T. Le, K. Shu, D. Lee, Authorship attribution for neural text generation, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online, 2020, pp. 8384–8395. URL: https://aclanthology.org/2020.emnlp-main.673. doi:10.18653/v1/2020.emnlp-main.673.

[6] OpenAI, GPT-4 Technical Report, Technical Report, OpenAI, 2023.

[7] E. Clark, T. August, S. Serrano, N. Haduong, S. Gururangan, N. A. Smith, All that's 'human' is not gold: Evaluating human evaluation of generated text, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Association for Computational Linguistics, Online, 2021, pp. 7282–7296. URL: https://aclanthology.org/2021.acl-long.565. doi:10.18653/v1/2021.acl-long.565.

[8] I. Solaiman, M. Brundage, O. Jack, C. Openai, A. A. Openai, A. Herbert-Voss, J. W. Openai, A. R. Openai, G. K. Openai, J. Wook, K. Openai, S. Kreps, M. M. Politiwatch, A. Newhouse, J. Blazakis, K. Mcguffie, J. Wang, Release Strategies and the Social Impacts of Language Models, Technical Report, OpenAI, 2019. URL: https://arxiv.org/abs/1908.09203v2. arXiv:1908.09203.

[9] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. Mccandlish, A. Radford, I. Sutskever, D. Amodei, Language Models are Few-Shot Learners, in: Advances in Neural Information Processing Systems, volume 33, 2020, pp. 1877–1901. URL: https://commoncrawl.org/the-data/.

[10] M. Orabi, D. Mouheb, Z. Al Aghbari, I. Kamel, Detection of Bots in Social Media: A Systematic Review, Information Processing & Management 57 (2020) 102250. doi:10.1016/J.IPM.2020.102250.

[11] R. Mohawesh, S. Xu, S. N. Tran, R. Ollington, M. Springer, Y. Jararweh, S. Maqsood, Fake Reviews Detection: A Survey, IEEE Access 9 (2021) 65771–65802. doi:10.1109/ACCESS.

2021.3075573.
[12] D. M. Katz, M. J. Bommarito, S. Gao, P. Arredondo, Gpt-4 passes the bar exam, Available at SSRN 4389233 (2023).
[13] A. M. Sarvazyan, J. Á. González, M. Franco Salvador, F. Rangel, B. Chulvi, P. Rosso, Overview of autextification at iberlef 2023: Detection and attribution of machine-generated text in multiple domains, in: Procesamiento del Lenguaje Natural, Jaén, Spain, 2023.
[14] S. M. Jiménez-Zafra, F. Rangel, M. Montes-y Gómez, Overview of IberLEF 2023: Natural Language Processing Challenges for Spanish and other Iberian Languages, Procesamiento del Lenguaje Natural 71 (2023).
[15] A. Uchendu, Z. Ma, T. Le, R. Zhang, D. Lee, TURINGBENCH: A benchmark environment for Turing test in the age of neural text generation, in: Findings of the Association for Computational Linguistics: EMNLP 2021, Association for Computational Linguistics, Punta Cana, Dominican Republic, 2021, pp. 2001–2016. URL: https://aclanthology.org/2021.findings-emnlp.172. doi:10.18653/v1/2021.findings-emnlp.172.
[16] X. He, X. Shen, Z. Chen, M. Backes, Y. Zhang, Mgtbench: Benchmarking machine-generated text detection, 2023. arXiv:2303.14822.
[17] J. Tourille, B. Sow, A. Popescu, Automatic detection of bot-generated tweets, in: Proceedings of the 1st International Workshop on Multimedia AI against Disinformation, MAD ’22, Association for Computing Machinery, New York, NY, USA, 2022, p. 44–51. URL: https://doi.org/10.1145/3512732.3533584. doi:10.1145/3512732.3533584.
[18] M. Mozes, M. Bartolo, P. Stenetorp, B. Kleinberg, L. Griffin, Contrasting human- and machine-generated word-level adversarial examples for text classification, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 8258–8270. URL: https://aclanthology.org/2021.emnlp-main.651. doi:10.18653/v1/2021.emnlp-main.651.
[19] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, P. G. Allen, RoBERTa: A Robustly Optimized BERT Pretraining Approach (2019). URL: https://arxiv.org/abs/1907.11692v1. doi:10.48550/arxiv.1907.11692. arXiv:1907.11692.
[20] J. Rodriguez, T. Hay, D. Gros, Z. Shamsi, R. Srinivasan, Cross-domain detection of GPT-2-generated technical text, in: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Seattle, United States, 2022, pp. 1213–1233. URL: https://aclanthology.org/2022.naacl-main.88. doi:10.18653/v1/2022.naacl-main.88.
[21] A. Estes, N. Vedula, M. Collins, M. Cecil, O. Rokhlenko, Fact checking machine generated text with dependency trees, in: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track, Association for Computational Linguistics, Abu Dhabi, UAE, 2022, pp. 458–466. URL: https://aclanthology.org/2022.emnlp-industry.46.
[22] E. Ferracane, S. Wang, R. Mooney, Leveraging discourse information effectively for authorship attribution, in: Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Asian Federation of Natural Language Processing, Taipei, Taiwan, 2017, pp. 584–593. URL: https://aclanthology.org/I17-1059.

[23] J. Shao, A. Uchendu, D. Lee, A reverse turing test for detecting machine-made texts, in: Proceedings of the 10th ACM Conference on Web Science, WebSci '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 275–279. URL: https://doi.org/10.1145/3292522.3326042. doi:10.1145/3292522.3326042.

[24] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, C. Finn, Detectgpt: Zero-shot machine-generated text detection using probability curvature, 2023. arXiv:2301.11305.

[25] V. Macketanz, B. Naderi, S. Schmidt, S. Möller, Perceptual quality dimensions of machine-generated text with a focus on machine translation, in: Proceedings of the 2nd Workshop on Human Evaluation of NLP Systems (HumEval), Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 24–31. URL: https://aclanthology.org/2022.humeval-1.3. doi:10.18653/v1/2022.humeval-1.3.

[26] S. Hochreiter, J. Schmidhuber, Long Short-Term Memory, Neural Computation 9 (1997) 1735–1780. doi:10.1162/neco.1997.9.8.1735.

[27] L. Breiman, Random Forests, Machine Learning 45 (2001) 5–32.

[28] P. Przybyła, Detecting Bot Accounts on Twitter by Measuring Message Predictability, in: L. Cappellato, N. Ferro, D. E. Losada, H. Müller (Eds.), Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum, CEUR-WS.org, Lugano, Switzerland, 2019. URL: http://ceur-ws.org/Vol-2380/paper{_}58.pdf.

[29] A. Holtzman, J. Buys, L. Du, M. Forbes, Y. Choi, The Curious Case of Neural Text Degeneration, in: 8th International Conference on Learning Representations, ICLR 2020, OpenReview.net, Addis Ababa, Ethiopia, 2020. URL: https://openreview.net/forum?id=rygGQyrFvH.

[30] V. Sanh, L. Debut, J. Chaumond, T. Wolf, DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, in: Proceedings of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019, Vancouver, Canada, 2019. URL: https://arxiv.org/abs/1910.01108.

[31] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language Models are Unsupervised Multitask Learners (2018). URL: https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf.

[32] A. G. Fandiño, J. A. Estapé, M. Pàmies, J. L. Palao, J. S. Ocampo, C. P. Carrino, C. A. Oller, C. R. Penagos, A. G. Agirre, M. Villegas, MarIA: Spanish Language Models, Procesamiento del Lenguaje Natural 68 (2022). URL: https://upcommons.upc.edu/handle/2117/367156{#}.YyMTB4X9A-0.mendeley. doi:10.26342/2022-68-3.

[33] Y. Lin, J.-B. Michel, E. Aiden Lieberman, J. Orwant, W. Brockman, S. Petrov, Syntactic annotations for the Google Books NGram corpus, in: Proceedings of the ACL 2012 System Demonstrations, Association for Computational Linguistics, Jeju Island, Korea, 2012, pp. 169–174. URL: https://aclanthology.org/P12-3029.

[34] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, PyTorch: An Imperative Style, High-Performance Deep Learning Library, in: H. M. Wallach, H. Larochelle, A. Beygelzimer, F. D'Alché-Buc, E. B. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver,

BC, Canada, 2019, pp. 8024–8035. URL: https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

[35] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, A. M. Rush, Transformers: State-of-the-Art Natural Language Processing, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics, Online, 2020, pp. 38–45. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[36] D. P. Kingma, J. L. Ba, Adam: A method for stochastic optimization, in: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, ICLR, San Diego, USA, 2015. URL: https://arxiv.org/abs/1412.6980v9. arXiv:1412.6980.

[37] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[38] M. Honnibal, I. Montani, S. Van Landeghem, A. Boyd, spaCy: Industrial-strength Natural Language Processing in Python (2020). doi:10.5281/zenodo.1212303.