UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Dept. of Computer Systems and Computation

Machine-Generated Text Detection and Attribution

Master's Thesis

Master's Degree in Artificial Intelligence, Pattern Recognition and Digital Imaging

AUTHOR: Sarvazyan, Areg Mikael

Tutor: Rosso, Paolo

External cotutor: FRANCO SALVADOR, MARC

ACADEMIC YEAR: 2022/2023

Abstract

The strong language capabilities of current Large Language Models (LLMs) are motivating a large-scale adoption in the workflows of businesses and individuals. These LLMs have the potential to be used in cutting-edge applications, but they could also be leveraged for malicious intents. A promising line of research to control their use is detecting Machine-Generated Text (MGT) and attributing it to specific models. While this approach has been explored for specific domains, the detection and attribution of MGT in more realistic scenarios is yet to be studied. This work explores MGT detection and attribution in a multilingual, multi-domain, multi-style and multi-generator setting. This is achieved by compiling corpora for the tasks, organizing an evaluation campaign, and developing various MGT detection and attribution models. Our focus is threefold: (i) to study the MGT detection and attribution models that achieve high performance under various settings, (ii) to explore their generalization capabilities to different scenarios, analyzing their applicability to unseen types of data, and (iii) to successfully organize an evaluation campaign for MGT detection and attribution, thereby encouraging further developments in this line of research.

Resumen

Las sólidas capacidades lingüísticas de los Modelos de Lenguaje de Gran Tamaño (LLMs, por su sigla en inglés) actuales están motivando su adopción a gran escala en los flujos de trabajo de empresas y particulares. Estos LLMs tienen el potencial de ser utilizados en aplicaciones de vanguardia, pero también podrían ser aprovechados con intenciones maliciosas. Una línea prometedora de investigación para controlar su uso es la detección de Texto Generado por Máquina (MGT, por su sigla en inglés) y su atribución a modelos específicos. Si bien este enfoque se ha explorado para dominios específicos, es escaso el estudio de la detección y atribución de MGT en escenarios más realistas. Este trabajo explora la detección y atribución de MGT en un entorno multilingüe, multidominio, multiestilo y multimodelo. Esto se logra mediante la compilación de conjuntos de datos para las tareas, la organización de una campaña de evaluación y el desarrollo diversos modelos de detección y atribución de MGT. Nuestro enfoque es triple: (i) estudiar los modelos de detección y atribución de MGT que logren un alto rendimiento en diversos contextos, (ii) explorar sus capacidades de generalización a diferentes escenarios, analizando su aplicabilidad a tipos de datos no observados, y (iii) organizar con éxito una campaña de evaluación para la detección y atribución de MGT, fomentando así nuevos desarrollos en esta línea de investigación.

Acknowledgements

I wish to sincerely thank all those who have supported and guided me throughout the journey of completing this thesis, without whom this work would not have been achievable. Their collective efforts have played an instrumental role in shaping this work and enriching my academic experience.

Firstly, I extend my thanks to all who contributed toward making me a better researcher and professional. My advisors, Drs. Paolo Rosso and Marc Franco Salvador, who trusted, supported, and guided me through the correct avenues of research that resulted in this work. The research team at Symanto Research for their contributions through research questions, discussions, ideas, and guidance in natural language processing research. Specifically, José Ángel González Barba for sharing his extensive research and scientific writing knowledge as a co-author in the contributions generated from this work; Angelo Basile for his support in techniques to understand text datasets in this work, and participating in human annotations; Mara Chinea Rios for a custom implementation of a specific baseline; as well as Guillermo Pérez-Torró and Ian Borrego Obrador for insightful discussions, and their participation in human annotations. Drs. Francisco Rangel and María Berta Chulvi Ferriols for the help in organizing the shared task that is a major part of this work.

I would also like to thank my closest friends, and my parents Irina and Mikayel, all of who were always there for me, heard my voice in good and bad times, and supported me throughout. Finally, I thank my dog roommate Guido, for always brightening my mood.

Contents

List of Figures

List of Tables

Introduction 1

In this introductory chapter we introduce the objectives of this work, while also mentioning the articles and academic presentations that resulted from it. More specifically, we provide motivation for studying Machine-Generated Text (MGT) detection and attribution, describing the framework and goals of the thesis. Lastly, we present how the contents of this work are organized.

1.1 Motivation

The recent surge in capabilities of Large Language Models (LLMs) is motivating a large-scale adoption in the workflows of businesses and individuals. Tasks like creative writing, coding, or information seeking through search services are nowadays aided with LLMs to reduce human effort. The impact of LLMs on society is not negligible, and it has been estimated that the adoption of LLMs could affect to at least 10% of the tasks performed by 80% of the workforce in some countries [46].

In addition, there have been recent large-scale democratization efforts to broaden the public's access to large models [147, 165, 180, 150] which, paired with substantial access to high-end computing power through cloud services and APIs, have made it easier for non-technical people to interact with and use LLMs for various interesting applications [46, 99].

These generative models have the potential to be used in cutting-edge applications. Prominent examples include ChatGPT [118], which set the record for the application with the fastest-growing user base,[1] programming assistants such as Codex [20], or to support financial advisors [183].

However, efforts in AI democratization have also lowered the barrier of entry for users to generate high-quality, multi-style and multi-domain text in a massive scale. This means that motivated users could leverage LLMs for malicious intents, e.g., spreading propaganda or disinformation by generating human-like fake news [59], opinions [32], or scientific papers [73], posing a threat to the reputation of companies,

[1] `https://www.reuters.com/technology/chatgpt-sets-record-fastest-growing-user-base-analyst-note-2023-02-01/`

academic institutions and individuals [82, 80]. Moreover, the aforementioned advancements have also promoted discussions in ethical AI [177] as well as model, data and training regulations,[2] and new licenses [11, 28]. Additionally, the public's usage of LLMs may compromise data quality. Recent research has shown that 33% to 46% of manually-annotated data may have been generated by an LLM [170]. In addition to this, Shumailov et al. [155] argue that future LLMs will suffer from *model collapse*, a phenomenon that manifests when models are trained on an overwhelmingly larger amount of generated data. In general, these findings suggest that future LLM-powered applications may (i) model language that is insufficiently diverse and complex, given that the more generated data they observe, the less they will be able to model the intricacies of real language, and (ii) lack domain-specific knowledge, further perpetuate biases or memorize content, since generated data often reflects content that is readily available and widely consumed, meaning that an overwhelming majority of generated content would focus on popular topics and opinions, resulting in models that may not be sufficiently exposed to specific domains or opinions.

Content moderation and defense against large-scale spam, propaganda or disinformation attacks; new developments in AI regulations and licenses; as well as the need to guarantee and maintain high-quality language data are strong motivators to ensure a responsible use of LLMs and generated content in general. A promising approach to carry this out consists in detecting MGT, and applying content moderation techniques on top. In this line of research, there has been a recent surge of models [107], services [117], and watermarking techniques [84], aimed towards detecting or assisting to detect MGT. This approach has been explored in specific scenarios, including detecting fake news [186], bots in online environments [164], and MGT in technical research [138]. Furthermore, there has been a recent focus toward larger-scale studies in MGT detection techniques in various languages and domains through evaluation campaigns [81, 152]. However, from the legal, security, and forensic points of view, solely detecting whether a text has been automatically generated is not good enough to identify the actors and motives behind the MGT. In that sense, model attribution [167] can be employed to attribute a text to a specific LLM or families of LLMs, yielding more insights into the party generating potentially malicious MGT.

This work is an attempt to further advance the research barriers into a better understanding of the tasks of MGT detection, where one must identify whether a text has been generated automatically or written by a human, and MGT attribution, where once an MGT has been correctly identified, one predicts the model that generated it.

[2]European Commission, Proposal for a Regulation of the European Parliament `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A52021PC0206`

		Pre-trained	Fine-tuned
Accessibility	Not modified by a human	Everyone	Only technical
Computational Resources		Low	High
Human Resources		Low	Low
Generation Scale		High	High
Generation Quality		Medium	High
Accessibility	Modified by a human	Everyone	Only technical
Computational Resources		Low	High
Human Resources		High	High
Generation Scale		Low	Low
Generation Quality		High	Perfect

Tab. 1.1.: Types of MGT. In this work we focus on generations from pre-trained models without human modification.

For this, we curate AuTexTification 2023,[3] a high quality multi-domain and multi-style dataset as a benchmark for MGT detection and attribution systems. In compiling this dataset we consider the most accessible way for users to generate text (see Table 1.1), which involves little computational and human resources, and can be used to generate vast amounts of MGT. We leverage this dataset in the AuTexTification 2023 Shared Task with the purpose of exploring many dimensions of various approaches to solve these tasks, while also carrying out an extensive experimentation to study (i) the generalization of the more popular approaches to new model families and parameter scales, as well as (ii) the feasibility of model family and parameter scale attribution, instead of fine-grained MGT attribution. For the following sections of this work, we will use *family* to refer to groups of models that share the same underlying architecture and are trained with the same data in the same manner.[4] Similarly, we use *scale* to refer to models of different families but with similar number of parameters.

1.2 Framework

This work has been carried out while the author was employed as a Junior Research Scientist at Symanto,[5] and is also supported by the valgrAI[6] - Valencian Graduate School and Research Network of Artificial Intelligence and the Generalitat Valenciana, and co-founded by the European Union.

[3]https://sites.google.com/view/autextification
[4]For instance, BLOOM refers to the family consisting of BLOOM-1b7, BLOOM-3b, BLOOM-7b1, etc.
[5]https://www.symanto.com/
[6]https://valgrai.eu/

1.3 Goals

Building upon the motivation of this work, we define the following goals:

- To compile a multilingual, multi-domain, multi-style and multi-generator dataset of human and generated text to evaluate MGT detectors and attributors in realistic scenarios.
- To promote research in MGT detection and attribution by successfully organizing a shared task.
- To study MGT detectors' generalization capabilities to different scenarios.
- To explore the feasibility of different approaches to tackle the open problem of MGT attribution.

1.4 Document Structure

The following chapters of this work are structured as follows. Chapter 2 provides a historical overview of text generation approaches, delving deeper into state-of-the-art methods and techniques that make these models stand out. Similarly, Chapter 3 describes the different approaches to MGT detection, also discussing the lack of research in MGT attribution. Moreover, Chapter 4 presents the AuTexTification 2023 dataset and shared tasks, including the data gathering process and evaluation of generations, the participating submissions and analysis of their results. Building on top of this, in Chapter 5 we carry out further experiments and analyses into the generalization capabilities of MGT detectors to unseen model families and parameter scales, and explore the feasibility of MGT attribution to model families and parameter scales in Chapter 6. Finally, Chapter 7 provides a summary and conclusions of the work, while also discussing future avenues of research in this field.

Text Generation 2

2.1 Introduction

Text generation, more generally sequence generation, has been an important line of research throughout many decades encompassing artificial intelligence. By generating coherent and contextually relevant text, Language Models (LMs) can facilitate tasks such as machine translation, summarization, question answering, dialogue, and open-ended content generation, tasks that are vital for improving human-computer interaction as well as reducing or removing barriers of communication.

In this chapter, we provide a historical overview of Natural Language Generation (NLG) models, and delve deep into the inner-workings of state-of-the-art NLG models that dominate most tasks in the field of Natural Language Processing (NLP). Specifically, we will describe (i) how and why the state-of-the-art models perform better than others, (ii) the various architecture, data, and speed improvements that were carried out to reach better capabilities, (iii) their generation process and common usage patterns, as well as (iv) evaluation metrics for open-ended generation.

2.2 History

2.2.1 First Steps

NLG research has been carried out for over 60 years prominently. The main focus was on building machines capable of generating text as a way to build assistive technology, improve and remove barriers of communication, and for creating natural language interfaces and dialogue systems that could aid and improve upon our day-to-day lives. As more and more advances have been developed, we have seen this technology be utilized by hundreds, thousands and now millions of users across the globe.

However, language generation can be dated back to Ramon Llull's Ars Magna, created in the 13th century. This was a system of logic based on general principles that was used in early Spain to prove statements about the christian God, which worked by spinning concentric paper-craft wheels to combine into letters. Yet, it was seven centuries later where focus on sequence generation and modelling became of importance through the works of Markov, who introduced the Markov chain [102]. This technique is now used in countless algorithms and is applied in various domains with different types of sequences.

A completely different but nevertheless also groundbreaking idea was presented by leaders of the Dadaist movement, who proposed an algorithm to make a dadaist poem, consisting in shaking and retrieving cut out words of newspapers [94]. While at the time this was part of an art movement, it defines a set of random processes that approximate some of the early techniques to generate language, since by cutting words and shaking them in a container, words that appeared more frequently would be more favored. This process essentially equates to unigram sampling as it is defined today.

Moreover, one of the first works focusing exclusively on NLG was carried out in Goldman [60], where the authors described the implementation of Conceptual Dependency networks, "unambiguous, language-free representations of meaning". This was the starting point that defined one of the main ideas behind NLG research: abstract representations that can capture meaning at various dimensions. As more efficient and adequate representations were designed, NLG systems were used in more and more applications.

2.2.2 Statistical Language Models

The first popular models are now known as statistical language models [139]. They were made up of various tables that kept track of various counts, which were employed to model language in a probabilistic setting. Once these tables had learned an adequate probability distribution of the language, one could generate text by sampling from them. These sampling procedures are omitted for statistical LMs in this work. However, similar sampling ideas are currently being used in state-of-the-art NLG models, which will be briefly described in the following sections.

More formally, assuming a vocabulary Y, in order to generate realistic and diverse language, we wish to find a model $\mathcal{M}$ that better approximates the probability of a sequence, $P_{\mathcal{M}}(y_0, \ldots, y_k)$ or the probability of the next symbol $y_k \in Y$ given a previously observed sequence $y_{<k} = y_0, \ldots, y_{k-1} \in Y^k$ of k symbols[7] $P_{\mathcal{M}}(y_k \mid$

[7]Due to the nature of most language systems, left-to-right was adopted by convention. However, it is important to note that this choice is arbitrary.

$y_0, \dots, y_{k-1})$. Assuming $P_{\mathcal{M}}$ models language adequately, by simply sampling from it we would be able to generate realistic phrases.

The relationship of the two probability expressions can be found through repeated applications of the chain rule:

$$P_{\mathcal{M}}(y_0, \dots, y_k) = P_{\mathcal{M}}(y_0) \prod_{i=1}^{k} P_{\mathcal{M}}(y_i \mid y_{<i}) \tag{2.1}$$

$$= P_{\mathcal{M}}(y_0) \prod_{i=1}^{k} P_{\mathcal{M}}(y_i \mid y_0, \dots, y_{i-1}). \tag{2.2}$$

Researchers quickly realized that approximating such a function was intractable and would give rise to numerous issues due to data and hardware constraints. Thus, many approximations were proposed. The first popular attempts to generate language were built upon the Markov assumption, consisting in limiting the observations to a fixed window by proposing n-gram LMs. These would approximate the conditional probability by constraining the history to a sequence of smaller length

$$P_{\mathcal{M}}(y_k \mid y_0, \dots, y_{k-1}) \approx P_{\mathcal{M}}(y_k \mid y_{k-n-1}, \dots, y_{k-1}) \tag{2.3}$$

and could easily approximate $P_{\mathcal{M}}$ by counting n-gram occurrences in a corpus. However, given the length constraints, these models were usually unable to generate text of high-enough quality, lacking especially in capabilities to maintain long range dependencies and avoiding generations of repeating sequences [43]. Another issue in their use was the need for smoothing techniques to take into account unseen n-grams, for which many heuristics were designed and evaluated [115, 21].

A different approach to model $P_{\mathcal{M}}$ was through Probabilistic Context-Free Grammars (PCFGs) [23, 89]. These were finite state machines with additional sets of rules that modeled language using paths of transitions through various states. Each transition modeled the likelihood of choosing that rule during the generation process, and was learned by counting the number of times that paths included its starting and ending states in the training process, by observing a corpus of text.

These two ideas were surpassed by Hidden Markov Models (HMMs) [8, 77], non-deterministic Markov chains that can incorporate n-gram relationships and additional more complex ones similar to those from PCFGs. However, while HMMs were not as expressive as PCFGs, they also did not suffer from more computationally expensive training, making them better alternatives for modeling and generating language.

Throughout the years PCFGs, HMMs and n-gram LMs were used in various generative (and otherwise) applications. Yet, after observing further developments in neural-based modelling techniques, researchers started focusing on neural models for generation, by first generating without considering previous history, and slowly introducing it in the generation process as hardware got better and faster.

2.2.3 Neural Language Models

The first neural LM was proposed by Bengio et al. [10]. It was a simple architecture consisting in an embedding matrix that mapped words to a vector representation to be used by a Multi-Layer Perceptron (MLP) to generate the next token. While the originally proposed architecture was slightly different, most popular versions can be formalized as variations of

$$\begin{aligned} y_k &\sim P_{\mathcal{M}}(y_k \mid y_{<k}) \qquad (2.4)\\ &\sim \mathrm{softmax}(We_{<k} + b), \qquad (2.5)\end{aligned}$$

with W and b being the weights and biases of the MLP, and $e_{<k}$ denoting the embedding of the previous context. The latter was usually obtained by concatenating the product of a one hot representation of each word, in a window of n previous words, with the embedding matrix E

$$e_{<k} = \left[E\mathbb{I}(y_{k-1}); E\mathbb{I}(y_{k-2}); \ldots ; E\mathbb{I}(y_{k-n+1})\right], \qquad (2.6)$$

where we denote the indicator function with $\mathbb{I}$.

This way, the model learned $P_{\mathcal{M}}(y_k \mid y_{<k})$ by observing large quantities of texts, with the added advantage that since it operated in a continuous space, it learned a continuous function (as opposed to learning a table of counts of occurrences), meaning that its generalization capabilities to unseen contexts was better. It was the first LM trained using backpropagation [140], and it laid the foundations of current state-of-the-art methods in doing so. However, this model and its further modifications were constrained in a similar manner to n-gram LMs by the length of the input sequence, and they lacked the capability for learning long-range dependencies.

Hence, Recurrent Neural Networks (RNNs) [45] were considered instead of MLPs. Here, the instead of only producing outputs based on the inputs alone, the LM also includes a hidden state or memory which also conditions the result. Formally, this only resulted in the inclusion of an additional term in (2.4),

$$P_{\mathcal{M}}(y_k \mid y_{<k}) = \mathrm{softmax}(We_{<k} + Vh + b), \qquad (2.7)$$

where we also introduce a hidden vector h and its corresponding weight matrix V. This way, the model maintained a memory of previous elements in the sequence without having to explicitly observe them multiple times. However, in introducing cycles it added complexity to the training dynamics of these models, for which *backpropagation through time* [109] was designed, consisting in an unrolling of the network up to a fixed length to compute gradients and successfully learn to model language. In addition, exact gradient methods [179] were developed but found to be computationally too expensive to be feasible in practice.

While recurrent networks have some advantages for sequence tasks, the conditionality introduced by the cyclic connections also comes with the gradient vanishing and explosion problems especially when layered to obtain deep networks. When unrolling an RNN, longer paths from input to output can cause the gradient to explode or vanish. This occurs because a gradient much greater than 1 leads to increasingly larger gradients in subsequent modules, destabilizing training. Conversely, a gradient close to 0 diminishes until it becomes insignificant, resulting in no parameter updates. These issues were reduced by modifying the RNN architecture, resulting in new recurrent neural units such as the Long Short-Term Memory [70], the Gated Recurrent Unit [22], the Simple Recurrent Unit [92, 119] and more. Additionally, some architectures included bidirectional RNNs [148] to obtain direction-invariant representations, and *attention* [4] layers were introduced, responsible for managing information flow. The latter then became a key component in state-of-the-art LMs, as will be mentioned in future sections. Interestingly, while these models are not as popular for language modeling today, there are currently many efforts to re-introduce recurrences within state-of-the-art LM architectures. Prominent examples include RWVK [124] and Recurrent Memory Transformers [18].

Another idea consisted in expanding upon MLP LMs by considering larger and larger context windows instead of adding memory units to the architecture. This was achieved through Convolutional Neural Networks (CNNs), which include various learned kernels that, when multiplied with the input, act as signal filters by enhancing important information and weaken the noise. CNNs are implemented by replacing the free-form W in (2.4) with a circulant matrix where all rows include the same values and are rotated by one position. Through layers of CNN modules and faster implementations using Fourier transforms, these models learned more adequate representations of language. Note that unlike more popular applications of CNNs such as for image data, pooling was typically not used in these models to maintain positional information. Notable examples of CNN LMs include Gated CNNs [35] and lightweight CNNs [182]. Similarly to RNNs, these ideas are being used and incorporated in state-of-the-art architectures, some of which include H3 [53] and Hyena [130] which break records in long-range tasks [162].

Finally, as better hardware was developed, more data could fit in GPUs and more computation could be carried out faster. This meant that we could do away with the need of memories altogether as in RNNs, and stop considering only small windows of the text as in CNNs, and instead process the complete context at once to model $P_{\mathcal{M}}$. This is the main idea of the Transformer [169], which is comprised solely of attention and MLP layers, and together with a few tricks to improve the optimization process [74], is now the main idea behind current language models, becoming state-of-the-art not only in NLG and NLP in general [17, 97], but also in other areas that involve different types of signals [42, 131, 78].

2.3 Decoder-only Transformers

The emergence of Large Language Models (LLMs) has had a profound impact on our society, revolutionizing various aspects of our lives. These advanced AI systems have significantly enhanced natural language understanding and generation, resulting in (i) more accurate and sophisticated applications, (ii) effective communication across different cultures through accurate translations, (iii) efficiency in creative tasks by using them as writing assistants, (iv) improved user experience through dialogue systems. The LLMs used for these tasks consist mainly in decoder-only Transformers (i.e. models that only attend to past inputs), as opposed to the original Transformer architecture which was an encoder-decoder model. Researchers have built upon the original architecture, adding more layers of different types and including better training and inference schemes. While encoder-decoder models can and are also used for NLG [135, 26], their larger parameter scales make them more computationally expensive and less favored in practice, with the best encoder-decoder models not achieving comparable capabilities to decoder-only models. In this section we will briefly explain how the decoder-only Transformer works, describe the present techniques that establish this architecture as state-of-the-art, along with the ideas that make LLMs what they are. We will also present various decoding strategies used to generate text, as well as various techniques to evaluate generations.

2.3.1 Architecture

Decoder-only Transformer LMs, also known as autoregressive or causal LMs, are particularly popular at time of writing. As w, they are composed of (i) an embedding layer, which converts discrete tokens to a learned vector representation that considers both meaning and position, (ii) a stack of Transformer decoder layers, (iii) an unembedding to get a final representation of the possible tokens to emit, and (iii) a final MLP with a softmax activation, which accurately models P_M.

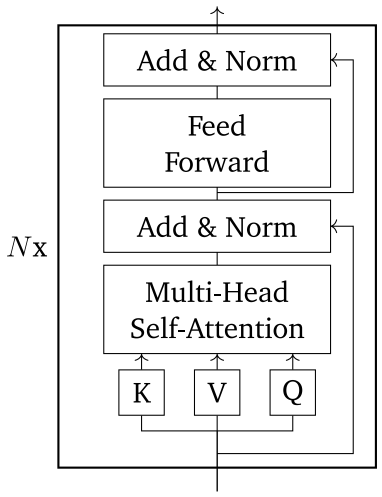


Fig. 2.1.: Transformer decoder as presented in [169].

While the first and last component are common in most neural LMs, it is the second component consisting in a stack of Transformer decoders that allows these models to learn better language representations. Figure 2.1 presents a decoder-only Transformer, which consist in MLPs, multi-head self-attention, layer normalization [3], and residual connections [68]. We will briefly describe how the latter three are defined and what their role is in the network. For more information, please refer to the original Transformer paper [169].

Multi-Head Self-Attention. It is a technique that controls the flow of information in a complex manner, and is the main mechanism that enhanced language modelling capabilities in the last few years. Transformer LMs use a particular type of attention: *scaled dot product attention*. In a decoder-only transformer, given an input sequence $x_1, \dots, x_n$ of n d-dimensional embeddings as rows of a matrix X, scaled dot product self-attention computes the dot product between elements of the sequence as a way to measure similarity scores, using these learned attention weights to adequately control how much information the different elements contribute to each other's representations. Traditionally, this is understood through the lens of information retrieval, where a set of queries Q are compared with a set of keys K, and these similarities are used as weights of their corresponding values V,[8]

$$a(Q, K, V) = \operatorname{softmax}\left(\frac{Q^T K}{\sqrt{d_h}}\right)V. \tag{2.8}$$

Here, we use d_h for the head dimension, and by dividing by its square root, we ensure that the resulting distribution from the softmax operation has a variance

[8]For example, when you search for videos in popular platforms, search engines map your query (text in the search bar) against a set of keys (video title, description, etc.) associated with candidate videos in their database, then present you the best matched videos (values).

of 1. Notably, the computation is split into various attention heads to employ smaller matrices, and with the idea that each of these will learn different patterns of information flow through the use of different learned weights,

$$\text{head}_i = a(W_i^Q X, W_i^K X, W_i^V X). \tag{2.9}$$

We concatenate the head outputs, projecting to the original dimension, to obtain the final output of multi-head attention,

$$mha(X) = [\text{head}_1; \ldots; \text{head}_h] W_O. \tag{2.10}$$

Layer Normalization. This is a common technique used to normalize the distribution of the vectors in a given layer, ensuring more stable training dynamics, usually applied after multi-head self-attention and MLP layers. The method directly estimates the mean μ^l and standard deviation σ^l of a layer $h^l = h_1^l, \ldots, h_H^l$ consisting of H embedding vectors,

$$\mu^l = \frac{1}{H}\sum_{i=1}^{H} h_i^l, \quad \sigma^l = \sqrt{\frac{1}{H}\sum_{i=1}^{H}(h_i^l - \mu^l)^2}, \tag{2.11}$$

and normalizes accordingly, also including a learned affine transform through g and b for better training:

$$\text{LayerNorm}(h) = \frac{g}{\sigma^l} \odot (h^l - \mu^l) + b. \tag{2.12}$$

Here, $\odot$ denotes the Hadamard product.

Residual connections. As larger models are trained, these tend to take longer to start training efficiently due to random initializations of the model parameters. This is what residual connections attempt to solve by introducing a simple sum of the input x, and the output of a layer $\mathcal{F}$,

$$\text{ResidualConnection}(x, \mathcal{F}) = \mathcal{F}(x) + x. \tag{2.13}$$

In this manner, when computing gradients through backpropagation, the path that goes through x always yields a gradient of 1, ensuring that the model parameters will be updated by optimizing the residual mapping instead of the original one.

Whilst this has been a brief overview of the components of the decoder-only transformer, it is only a simplified form of what current LLM architectures include. Additionally, given their proven success in countless pattern recognition tasks, many researchers have recognized the need to understand what makes them special. In the following sections we briefly explore research in decoder-only Transformer interpretability, and illustrate some model variants that have proven effective in LLM architectures.

2.3.2 Why Decoder-only Transformers Stand Out

These models have been getting bigger and better as more research has been carried out, and there has been many efforts to understand their learning dynamics and interpret what they have learned. The general consensus is that attention is the main mechanism that allows for their dominance with respect to other architectures.

Early work in attention interpretability has found that these models learn patterns that are familiar to us. Prominent examples include [44], where authors found that zero-layer attention-only networks (i.e. only the product of the embedding and unembedding matrices) approximate bigram models. Similarly, one-layered architectures learn skip-trigram models, while two-layered architectures learn to carry out an induction process, i.e. completing a previously seen pattern. This was the first step in understanding what decoder-only LMs are able to achieve currently by *in-context learning*, learning to carry out tasks by observing a small subset of examples in their input context. However, other works have found that attention is not the only important aspect in Transformers, and that deep attention-only networks struggle to learn more complex patterns [41].

Another avenue of interpretability consisted in understanding what the MLP layers' role was [58], finding that MLPs in Transformer-based LMs operate as key-value memories, with each key correlating with particular textual patterns in training examples, and each value being a distribution over the vocabulary. Moreover, there have been efforts in understanding their inference process. Prominent examples include [116, 9], techniques used to extract the tokens of highest confidence in the latent layers of decoder-only Transformer. Using these methods, authors found many cases where the model was very confident in one particular token from a set layer, but in some peculiar instances it moved its confidence to other tokens in the last layer.

Finally, large amounts of research has been carried out to understand the scaling laws of these models, in most cases finding that the more data and the bigger the model, the better its final performance will be [79, 71]. This avenue of research aids us in taking more informed business decisions and wasting little resources, given that we can now estimate how much compute and data is needed to train a language model to perform as desired [183].

In order to go from small models of at most 100 million parameters, to current LLMs such as GPT [134, 17], BLOOM [147] or LLaMa [165] that have billions of trainable weights, many important challenges had to be overcome. Specifically, we can group these in two categories: resource and data challenges. The former involved issues such as fitting larger models in multi-GPU systems, training them effectively and using them efficiently in inference mode, while the former refer to challenges regarding the quality and quantity of language data available for training. In the following sections we briefly describe the most important advances that helped the research community overcome these challenges.

2.3.3 Model Improvements

Various types of techniques have been investigated to improve the original Transformer architecture. While here we focus of decoder-only models, many of these techniques are also used in other types of architectures as well.

Better Positional Information. Since self-attention is permutation invariant by design, in order for Transformer models to learn linguistic relationships, they need to be given positional information. In the original architecture, this was done via a set of interweaving sinusoidal functions. Throughout the years many other techniques were considered, such as using learned [1] or relative [153] positional encodings, and the LLM researchers converged to rotary embeddings, RoPE [158]. Rotary embeddings encode positions via block diagonal matrices consisting of blocks of euclidean rotation matrices. In this manner, they include positional information as rotations in a high dimenisonal space, thus maintaining both absolute and relative relationships. RoPE embeddings were first introduced in LLMs through EleutherAI's GPT-J-6B[171] and GPT-NeoX-20B [14], and after observing their success, they were included by default in most LLM architectures as of today.

Different Circuits. Various modified configurations of the original transformer were considered. Specifically, researchers found that pre-normalizing (i.e., including layer normalization inside the residual blocks, before self-attention or MLP layers), stabilized training with respect to the original architecture [185]. Additionally, researchers considered a parallel circuit of the MLP and self-attention layers instead of a serialized one, finding that this alternative yielded similar results more efficiently [14]. There has also been discussion regarding whether tying or untying the embedding and unembedding matrices is better for LLM training. [9] Currently no agreement has been reached within the research community. Prominent examples include Pythia models [13] which do not tie the matrices, and PaLM models [24] that do.

[9]This is commonly known as *weight tying*.

Better Activations Throughout the years, there have been many discussions regarding the types of activation functions that perform better for different tasks in different neural architectures. Specifically, in decoder-only Transformer LLMs it has been found that SwiGLU activations introduced in PaLM models [24] perform better than others. These usually require more computational resources but yield better results in the long run.

Longer Contexts. Improving the context length of LLMs is crucial for long-range dependency tasks that are defined at document or multi-document levels. External memories [83] and the reuse of hidden states [33] were considered, but these efforts converged in ALiBi [132], where by simply adding constant terms in the query-key product, decoder-only models trained on 1024 context lengths were able to generalize to 2048 context lengths easily. Currently, ALiBi is employed in most architectures such as BLOOM [147] and MPT [163].

Decoder-only transformers have greatly benefited from these techniques, with many of them being used in most newly released LLMs. However, model improvements alone were not enough, hardware usage and data quality improvements were key too.

2.3.4 Resource Improvements

Three key types of innovations stand out regarding resource use and guaranteeing large models fit in memory and can be trained and used: parallelism, lower precision parameters, and fused kernels.

Parallelization. In order to efficiently train large models that can only fit in multiple GPUs, many proposals were made for different parallel schemes to take better advantage of hardware resources and train and infer more efficiently. These techniques are encompassed in three categories: (i) data parallelism, consisting in replicating the model across GPUs and feeding in different training examples, (ii) tensor parallelism, where model weights are sharded across GPUs, and (iii) pipeline parallelism, where the model layers are split between GPUs. Many of these advances have been incorporated into popular Transformer training and inference frameworks such as Megatron-LM [85], DeepSpeed [95], and HuggingFace Accelerate [66]. Some also include methods that offload optimization statistics to secondary memory [137], thus further democratizing billion-scale parameter LLM training.

Lower Precision. In the past, most models were trained with the popular FP32 single precision arithmetic proposed in the IEEE 754-2008. However, as models became larger, they required more memory to fit in single consumer-grade GPUs.

This issue motivated research toward lower-precision training and inference through quantization techniques, i.e. discretizing or reducing the representation ranges of model weights, thus requiring less bits-per-parameter to load the model in memory. Through various developments, it became standard for LLMs to be trained using mixed-precision FP16 formats [105] and new research suggests that other formats, e.g. 8-bit or 4-bit floats [106, 38], or even 8-bit or 4-bit integers [36, 184] might be favored in the future for more efficient training and inference of LLMs. Quantization techniques are offered through popular libraries such as bitsandbytes [37].

Fused Kernels. Another avenue for optimizing training and inference runtime consists in improving the implementation of the modules within Transformers. Here, instead of implementing new techniques using common deep learning frameworks, researchers are opting to invest time to implement efficient CUDA kernels and carrying out kernel fusion [172], i.e. combining various kernels into single ones, thus reusing loaded data and removing redundant loading and storing operations that usually occur between kernel calls. A prominent example of this is FlashAttention [34], a plug-in efficient implementation of scaled dot-product attention that is now used in all LLMs and has been natively incorporated in the newest PyTorch [122] version.

2.3.5 Data Improvements

While the aforementioned improvements allowed for efficient usage of resources for training and inference of LLMs, there was another dimension that needed to be addressed: data quality. The internet contains vast quantities of text data. However, most of it is not suitable for training language models in its raw form. Rather, it requires filtering, cleaning and further processing to be of high enough quality so that language models can better understand and generated language. Access to high quality multilingual text in many domains and styles posed many challenges that have been and are constantly worked on with the release of better and bigger corpora. Prominent examples include The Pile [56], ROOTS [90], and C4 [135], which are used in as training data in many current LLMs. Moreover, developments in instruction fine-tuning have also culminated in multiple released instruction datasets for fine-tuning existing pre-trained LLMs, such as P3 [142] and its multilingual version xP3 [111]. Finally, researchers have found that LLMs become better reasoners when also trained with code data.[10] Various models have also incorporated code datasets as part of LLM training [96], observing that in doing so these models perform better in various reasoning tasks.

[10]https://yaofu.notion.site/How-does-GPT-Obtain-its-Ability-Tracing-Emergent-Abilities-of-Language-Models-to-their-Sources-b9a57ac0fcf74f30a1ab9e3e36fa1dc1

Another aspect of language model data efficiency consists in the tokenization schemes used to split texts into tokens. It is crucial to carry this step out adequately. In doing so, one can reduce the vocabulary size into a smaller scale, thus both reducing the number of embedding and unembedding matrix parameters and also passing less possible information that the language model has to learn on (larger vocabularies imply more combinations of tokens that the model must observe to adequately model language). Efforts in better tokenization schemes consisted in splitting words into subwords [151, 87], reducing the vocabulary size and maintaining the ability to represent most words through simple concatenation. Further advances consisted in regularizing the splitting process to obtain different subword splits [133] and learning probabilistic tokenizers [86].

2.4 Text Decoding Strategies

Once a language model has been adequately trained, it is desirable to generate text. This is where *decoding strategies* are involved, which can be used to control the generation process such that it produces realistic and diverse text, or the most probable one. The latter usually consists in algorithms such as greedy decoding, consisting in always emitting the token with highest probability, or beam search, a tree-based search algorithm that attempts to find more probable texts. These are not usually applied for most NLG tasks, being mostly popular for tasks such as machine translation where obtaining the best possible translation is desirable [112]. Instead, for open-ended generation we will focus on decoding strategies that sample from $P_{\mathcal{M}}$ to produce realistic and diverse texts. The most commonly used ones are (i) top-k sampling [48], (ii) top-p or nucleus sampling [72], and (ii) contrastive search [160, 159].

The first two are sampling techniques, differing in the criterion used for sampling: top-k samples from the k tokens of highest probability, while nucleus sampling samples from the smallest set of tokens that comprise the top $p\%$ of probability mass. While top-k is able to filter ill-fitted tokens, since the algorithm consists in always selecting the k most probable tokens, it limits the model's creativity in flat distributions and maintains undesirable tokens in sharp distributions. Alternatively, nucleus sampling attempts to solve top-k's shortcomings by considering minimal sets of tokens that encompass most of the probability mass, thereby allowing the size of the tokens that can be sampled to increase or decrease dynamically according to the shape of $P_{\mathcal{M}}$. Importantly, in any sampling procedure there exists a temperature parameter that controls how peaky or flat the final $P_{\mathcal{M}}$ after applying the softmax in the LLMs head. Controlling this parameter is essential in obtaining diverse texts that maintain closeness to real human text, and inadequate values may result in highly repetitive, generic or otherwise random generations.

Finally, contrastive search builds upon these sampling techniques, incorporating a contrastive objective as part of the decoding strategy. This is done by linearly interpolating $P_{\mathcal{M}}$ of various candidates obtained through top-k sampling with a degeneration penalty measured as the maximal cosine similarity of the embedding representation of the token to be emitted together with the previous context's tokens. This way, larger degeneration penalties condition the decoding process to generate more similar text, while smaller ones allow for more diverse generations.

2.5 Evaluating Open-Ended Generation

Once generations have been produced, it is important to verify that these are of high quality. There are various aspects of text quality that must be evaluated, including token repetition and diversity, semantic coherence to the input, distribution similarity to human text, and more. In this section we describe the common text quality measures that are used in open-endedness research, and that will be used as a way to characterize generated text in this work. Specifically, we consider measures that only take into account the generated text, measures that compare generations to prefixes, and measures that contrast generated continuations with human continuations. We consider three types of sequences involved in generating MGT: the original human text y_h, the prefix or prompt used as input to an LLM y_p, and the generated continuation y_g.

One of the main ways to measure generated text is by calculating various statistics of generated text. These are measured using text perplexity, repetition of n-grams [175, 160] and diversity [175], ratios of stop-words and symbols, as well as Self-BLEU [190].

Perplexity. Given a probability distribution p, we can measure its perplexity as $2^H(p)$, where $H(p) = -\sum_{y_g \in Y^*} p(y) \log_2 p(y)$ is the entropy of p. To measure text perplexity, we can use $p \equiv P_{\mathcal{M}'}$, with $\mathcal{M}'$ being a different model to the one whose generations are being evaluated.

n-gram Repetition and Diversity. They were introduced as a way to measure the n-gram repetition in a text as well as its general diversity. Given a text y, rep_n is defined as

$$\text{rep}_n(y) = 100 \times \left(1.0 - \frac{|\text{unique } n\text{-grams}(y)|}{|\text{total } n\text{-grams}(y)|}\right), \tag{2.14}$$

and is usually obtained with $n \in \{2, 3, 4\}$, to be used to measure Diversity:

$$\text{diversity}(y) = \prod_{n=2}^{4} \left(1.0 - \frac{\text{rep}_n(y)}{100}\right). \tag{2.15}$$

Diversity measures text degeneration, where lower scores mean more severe degenerations. Generally, we hope that the generated text is not too diverse nor too repetitive.

Stop-word and Symbol Ratios. We can also obtain a good image of the quality of generated text by measuring the mean ratio of stop-words and symbols with respect to other tokens in the generated texts.

Self-BLEU. It evaluates the diversity of generated data using the known BLEU [121] metric used in machine translation tasks. This way, by measuring BLEU scores between generated sentences and averaging them, one obtains Self-BLEU scores for generated text.

Another approach to measure the quality of generated text is to measure its semantic similarity to the prefix used to generate it, or to measure the generation's entailment to the prefix. This can be measured by observing the number of entailed, neutral or contradictory generations with respect to the prompts, using a different language model trained for Natural Language Inference (NLI) [157].

In addition to comparing generated text statistics and similarities with the prefixes, we can also compare the generated text distribution with the human one using MAUVE [128]. This metric was proposed to measure the generations' distribution similarity to human text, by measuring an approximated KL divergence for these distributions, following a particular procedure to adequately sample from them. The authors approximate the human and generated text distributions in a discrete manner, by embedding the texts with an external LM and quantizing them through a clustering step into low-dimensional discrete representations.

2.6 Use and Misuse

With these technological and algorithmic advances, and the emergence of new capabilities of LLMs, there are now various ways to use and misuse these models. Prominent examples from the media regarding users misunderstanding how these models work, and their output's factual correctness, include lawyers that cited nonexistent cases,[11] or workers that share personal and company information.[12] In this section we aim to provide a brief overview of usage techniques and ways in which these models can be easily exploited to generate biased, toxic or generally harmful text.

In order to generate text using LLMs, one can simply generate continuations given a prefix, or use a prompt, i.e. text that instructs the model on what type of task it needs to perform. Various types of prompting techniques exist, and the need to understand how to effectively use them to control generations has created the new position of *prompt engineer* in several companies. Generally, several insights have been found: (i) it is better to be simple and specific, (ii) impreciseness should be avoided, (iii) whenever possible, it is desirable to include in-context examples of input-output pairs. We refer to Saravia [144] and White et al. [176] for a more complete overview of prompt engineering techniques.

Moreover, there have been findings that, through careful design, prompts and inputs in general can be used to deceive LLMs such that they generate undesirable outputs. Prominent examples of these include adversarial attacks [188, 173], where researchers evaluate an LLMs capability to be robust to small changes in the input; (ii) red teaming language models [55, 125], where researchers attempt to break these models through inputs that simulate power seeking, persuasion and other critical threat scenarios; (iii) prompt hacking to get the model to output what a user wants, through techniques such as prompt injection [63], jailbreaking [174] or prompt leaking [126] which manipulate the input in different ways, e.g. by prompting the model to ignore everything it was instructed and instead print the hidden prompt it was given at the start of the interaction. These techniques are important in that they allow for users to easily generate massive quantities of text that could be used for beneficial or malicious purposes.

[11]https://www.reuters.com/legal/transactional/lawyer-who-cited-cases-concocted-by-ai-asks-judge-spare-sanctions-2023-06-08/
[12]https://www.cnbc.com/2023/05/02/samsung-bans-use-of-ai-like-chatgpt-for-staff-after-misuse-of-chatbot.html

3

Detecting and Attributing Machine-Generated Text

3.1 Introduction

As was mentioned in Section 1.1, one of the ways to ensure a responsible use of LLMs is by detecting and attributing Machine-Generated Text (MGT). Detecting MGT has shown remarkable results under assumptions of identical domain distribution and access to the generative model. However, in cross-domain settings or with certain writing styles, only few works showed that the performance of specific detectors plummets [5]. This hints at a lack of generalization when these assumptions are broken.

MGT detection has become a more popular area of research, tripling its frequency in research publications in the last ten years.[13] This shows that the research community realizes how important of an issue MGT is, how much potentially harmful uses it has, and how identifying it from human text is paramount to mitigate harm. However, the field is still in its early stages: similar tasks such as MGT attribution have not been explored thoroughly yet. In the next sections we define both tasks and provide an overview of proposed solutions by various research works.

3.2 Detecting Machine-Generated Text

Formally, MGT detection is a task in which, given a text y, one must design a model $\mathcal{M}_D$ such that it can correctly identify whether y has been written by a human, or generated automatically. Specifically, when we discuss MGT in this work, we adopt the following definition: "Text that has been produced without human intervention". While other works use a more broad definition that may include human modifications, since our goal in this work is to focus in mitigating massive scale malicious generations, we opt to exclude this aspect due to the human resource cost (see Section 1.1 for a more in-depth discussion).

[13]According to `app.dimensions.ai`.

Generally, most works striving to detect MGT fall under the following perspectives: (i) machine-aided detection, (ii) zero-shot detection, and (iii) supervised detection. Additionally, there are preventive techniques that aim to watermark MGT such that it is easily identifiable automatically. In the following sections we briefly describe these approaches. For additional techniques and problems regarding MGT detection we refer to the survey carried out by Crothers et al. [31].

3.2.1 Machine-aided Detection

In this paradigm, a human detector of MGT is assisted with statistical methods that capture generation artifacts. Since humans are good at noticing incoherence or factual errors in text and automatic methods are good detecting statistical abnormalities of token distributions, machine-aided detection strives to leverage the best of both worlds. The most prominent example is GLTR [57], a suite of statistical tools that improve humans' detection rate of text generated by GPT-2 [134] from 54% to 72% without any training. Yet, it requires a significant amount of human effort to be useful in practical scenarios, e.g., preventing massive campaigns of disinformation.

3.2.2 Zero-shot Detection

These approaches usually work under the white-box assumption, where a defender has access to the text-generation model that generated an MGT. Then, the same model is used to detect texts generated by itself or similar models, focusing on log-probabilities of the generated tokens. Two prominent examples of zero-shot detectors are presented in [107] and [156]. In [156], an approach based on thresholding the sum of log-probabilities was found to detect MGT from a GPT-2 model with 85% accuracy. DetectGPT [107] built upon this idea, improving the zero-shot detection of fake news using log-probability ratios of text and perturbed samples. Zero-shot approaches are practical as they require no human intervention or training data. Nevertheless, their generalization capabilities to new generators are limited due to the white-box assumption, which severely constrains their application.

3.2.3 Supervised Detection

These detectors are trained in a supervised fashion using datasets consisting of human-authored and machine-generated texts. While some consist in more traditional approaches that consider features capturing frequency [54], fluency [30], style of texts [52], most are fine-tuned Transformer-based [169] language models such as RoBERTa [138, 156], BERT [75, 103] or GROVER+LogisticRegression [186], with results usually higher than 90% macro F_1 under in-domain and in-model scenarios.

Supervised detectors require diverse high-quality datasets that encompass various domains, text-generation models, generation hyper-parameters, and writing styles. However, the generalization capabilities of supervised detectors to new scenarios is still unexplored, and few works studied it tangentially for very specific detectors [5]. Depending on how well supervised detectors generalize, building these datasets could be impractical. To our knowledge, this is the first work that studies the generalization of Transformer-based supervised detectors across model families and parameter scales.

3.2.4 Watermarking

Instead of aiming to detect MGT, these techniques are designed to distinguish MGT from human-authored text by modifying the generator's decoding strategy. Thus, the MGT includes a signature that makes it easily identifiable as MGT for automatic detectors. A notable example is watermarking by randomly ranking logit scores [84], which ranks logit scores between tokens with a pseudo-random function, assigning them as part of the watermark through a set seed. Another interesting approach [64] is to use a multi-task learning framework, where the model learns a set of back-doors pre-defined by its owner. However, watermarking could encourage LLMs to generate lower quality text in an effort to satisfy watermark rules. Moreover, it requires enforcement, and malicious users could simply avoid using watermarked LLMs.[14] Lastly, recent efforts have shown that watermarking can be beaten via paraphrasing MGT with another non-watermarked LLM [141].

3.3 Attributing Machine-Generated Text

After carrying out MGT detection and identifying texts as MGT, from a forensic perspective the next step is to understand more about the actor behind the MGT. This includes carrying out MGT attribution, where a given MGT y needs to be attributed by a model $\mathcal{M}_A$ into one of many text generation models, i.e. the one that generated y. This is a similar to the problem of authorship attribution, where a model is tasked with identifying the human author that produced a given text. In human authorship attribution there have been vast amounts of research, taking into account various domains [143], models [120, 154], and languages [16]. However, this is not the case for MGT attribution. Moreover, various survey papers in the field of neural text forensics consider MGT attribution an open problem [31, 65, 166] that has not been thoroughly studied.

[14]At time of writing, the European Parliament is pondering watermark obligation for generative AI [12].

One of the first works in MGT attribution was from Uchendu et al. [167], where authors studied various human authorship attribution techniques as a first step to understanding the task. They tried various common machine and deep learning approaches, finding that random forests with linguistic and readability features performed the best. There also is a benchmark that includes MGT attribution in the list of tasks[168]. However, these works were carried out when much weaker text generation models than those we have today. The problem of attributing text to a generator was easier due to the high variability in writing signature, style and overall quality. That is currently not the case due to the better capabilities of newly-released LLMs (see Section 2.3). Moreover, these works studied MGT attribution in specific domains, but a more realistic study in multi-domain scenarios has not been carried out.

4 Evaluation Campaigns and Shared Tasks

4.1 Introduction

Evaluation campaigns have been at the core of the artificial intelligence research community for years. These initiatives are organized to encourage researchers, practitioners and enthusiasts to promote and push knowledge boundaries in particular research areas. They usually involve shared tasks, collaborative efforts where participating teams attempt to solve specific research problems, answering concrete questions regarding previously unexplored topics. Popular evaluation campaigns include SemEval [47], the Dialog State Tracking Challenge (DSTC) [178], the NIST Speaker Recognition Evaluation (SRE) [62], the Conference and Labs of the Evaluation Forum (CLEF) [7], and the Iberian Languages Evaluation Forum (IberLEF) [108]. The past years there have also been shared tasks to promote research in Machine-Generated Text (MGT) detection, prominent examples of which include the Detecting Automatically Generated Scientific Papers (DAGPap22) Shared Task [81], where participants developed systems to detect automatically generated scientific papers, and the Russian Artificial Text Detection (RuATD-2022) Shared Task [152], with the purpose of developing and studying MGT detectors of Russian text. In the following sections we present the datasets, systems and results of the AuTexTification 2023 Shared Task, which focuses in MGT detection and attribution in multi-domain settings.

4.2 The AuTexTification 2023 Shared Task

We find it important to promote research in MGT detection and attribution due to the presented motivation from Section 1.1, as well as the lack of research carried out to evaluate MGT detectors in realistic scenarios and the scarce work done for MGT attribution described in Chapter 3. This motivated us to organize the AuTexTification (**Au**tomated **Tex**t Iden**Tification**) 2023 Shared Task at IberLEF2023 (Iberian Languages Evaluation Forum 2023). This shared task is proposed to study: (i) the automatic detection of MGT, (ii) the generalization capabilities of MGT detectors to new domains, and (iii) the feasibility of fine-grained MGT attribution to one of many generation models. Furthermore, we automatically collected a multi-domain annotated dataset of human-authored text and MGT generated by various LLMs, which is a valuable resource for exploratory linguistic analysis of machine-generated and human-authored texts. For more details about this dataset, please refer to Section 4.3.2. To our knowledge, AuTexTification 2023 is the first shared task to study both MGT detection and attribution in a multi-domain setting for English and Spanish, while also focusing on generalization of MGT detectors to new domains.

The AuTexTification 2023 Shared Task includes two subtasks in English and Spanish in five different domains.

Subtask 1: MGT Detection. This subtask consists in distinguishing between human and generated text. It is framed as a binary classification task of human text (HUM) and MGT (GEN), where text from three domains is included in the training set, and submissions are evaluated in two unseen domains. This way, we aim to study the MGT detectors' cross-domain generalization capabilities.

Subtask 2: MGT Attribution. In this subtask, participants must attribute MGT to the model that generated it, out of six models. Thus, Subtask 2 is framed as a six-class classification task, where we strive to study the feasibility of fine-grained attribution. Differently to Subtask 1, the training and test splits include all five domains.

A total of 114 teams signed up to participate, of which 36 sent a total of 175 submissions, with 20 of them sending their working notes.

4.3 Dataset

4.3.1 Data Gathering Process

Having observed the need of a multilingual, multi-domain, multi-style and multi-generator dataset of MGT and human text, and in order to organize the AuTexTification 2023 shared task, we compile a dataset of human and generated texts that share the same prefix. For instance, given a human text "*Today it's 20 degrees. It is sunny in Valencia.*", we could use "*It is sunny in Valencia.*" as human text, and generate a continuation by prompting an LLM with "*Today it's 20 degrees.*". Therefore, both generated and human texts are plausible continuations of the same prefix and they can be compared fairly in terms of topics and domains. To build the dataset in this way, we opted for a data gathering process consisting in the steps depicted in

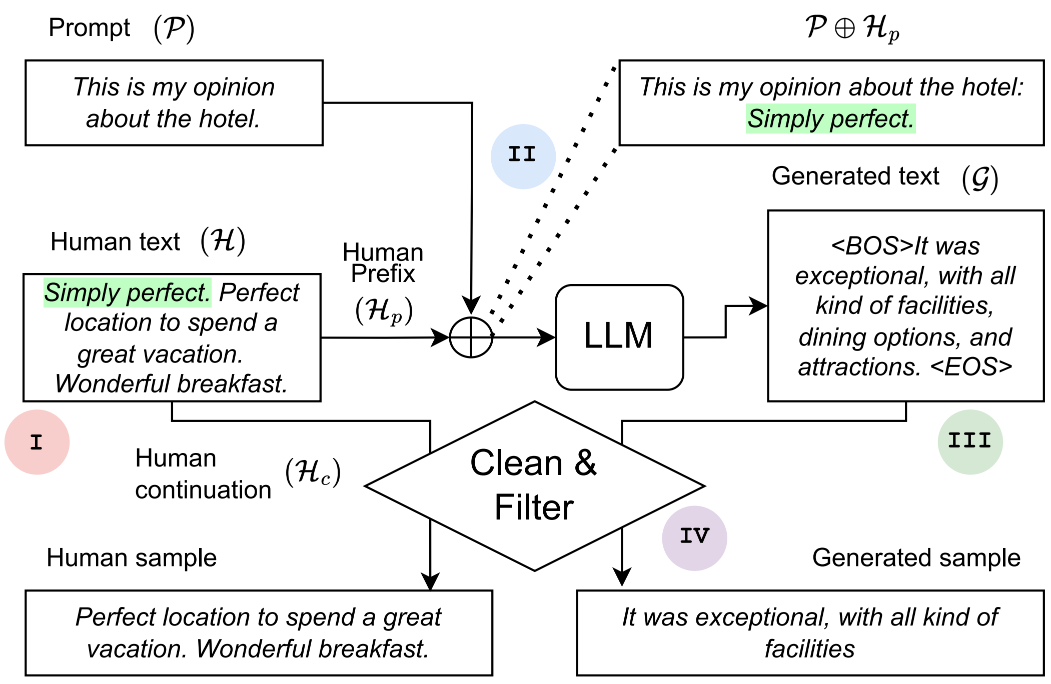

Prompt $(\mathcal{P})$
This is my opinion about the hotel.
$\mathcal{P} \oplus \mathcal{H}_p$
This is my opinion about the hotel: Simply perfect.
II
Generated text $(\mathcal{G})$
Human text $(\mathcal{H})$
Human Prefix $(\mathcal{H}_p)$
Simply perfect. Perfect location to spend a great vacation. Wonderful breakfast.
LLM
<BOS>It was exceptional, with all kind of facilities, dining options, and attractions. <EOS>
I
III
Human continuation $(\mathcal{H}_c)$
Clean & Filter
IV
Human sample
Generated sample
Perfect location to spend a great vacation. Wonderful breakfast.
It was exceptional, with all kind of facilities

Fig. 4.1.: Data gathering process.

Figure 4.1, namely (i) gathering human data, (ii) preparing the inputs for LLMs, (iii) generating MGTs, and (iv) cleaning and filtering the resulting texts.

We first gather a set of human-authored texts $\mathcal{H}$ from the source datasets for each domain and language. We manually analyze and define extraction schemes for splitting $\mathcal{H}$ into prefixes $\mathcal{H}_p$ and continuations $\mathcal{H}_c$ such that $\mathcal{H} = \mathcal{H}_p \oplus \mathcal{H}_c$. In some domains and source datasets, we also define prompts $\mathcal{P}$ to prevent the generation models from generating topic-inconsistent texts, e.g., guiding models to generate hotel reviews instead of car reviews by using a prefix from the COAH dataset, made up of hotel reviews. Afterwards, the prompts and prefixes $\mathcal{P} \oplus \mathcal{H}_p$ are fed into each LLM to obtain one resulting generation per prompt and prefix. We refer to the set of generations as $\mathcal{G}$. Texts from both $\mathcal{H}_c$ and $\mathcal{G}$ are fed into a text cleaning pipeline that removes duplicated spaces, multiple line breaks, and special symbols. Additionally, we ensure that the human continuation and generation obtained from the same prefix have roughly the same token-lengths by truncating to the minimum length of the two texts, thus removing token-length bias. Then, we apply a set of language identification filters: langdetect,[15] SpaCy FastLang[16] and fastText [76]. If one of these filters finds a text to be not in Spanish or English, the text is removed from our dataset.

[15]`https://github.com/Mimino666/langdetect`
[16]`https://github.com/thomasthiebaud/spacy-fastlang`

	English	Spanish
Legal	MultiEURLEX	MultiEURLEX
News	XSUM	MLSUM & XLSUM
Reviews	Amazon Reviews	COAR & COAH
Tweets	TSATC	XLM-Tweets & TSD
How-to	WikiLingua	WikiLingua

Tab. 4.1.: Source datasets containing human-authored text for the AuTexTification 2023 dataset.

To obtain the dataset for Subtask 1, we sample a subset of $\mathcal{H}_c$ labeled as HUM and a subset of $\mathcal{G}$ labeled with GEN. The dataset was then split into training and test sets for a cross-domain scenario: tweets, how-to articles and legal documents were included in the training set, while reviews and news data comprised the test set. To compile the dataset for Subtask 2, we only sample texts from $\mathcal{G}$, labeling each text with the LLM's name that generated it. The dataset is randomly split into training and test sets following 80%-20% proportions in a stratified manner. All the five domains are included in both training and test splits. The released version of the dataset for Subtask 2 includes anonymized model lables to remove bias towards particular models or model families in participating submissions.

4.3.2 The AuTexTification 2023 Dataset

The AuTexTification 2023 dataset consists of texts written by humans and LLMs in five domains: tweets, reviews, how-to articles, news and legal documents. These domains were chosen to encompass a range of writing styles, from more structured and formal to less structured and more informal. We collected human texts from publicly available datasets, namely: MultiEURLEX [19], XSUM [114], XLSUM [67], MLSUM [149], Amazon Reviews [104], WikiLingua [88], COAR & COAH [61], XLM-Tweets [6], TSATC [113], and TSD [93]. Table 4.1 groups these datasets per domain and language.

The MGT was generated from the human texts by using three BLOOM models [147], BLOOM-1B7,[17] BLOOM-3B,[18] and BLOOM-7B1;[19] as well as three GPT-3 models [17, 118]: babbage, curie, and text-davinci-003, with 1b, 6.7b and 175b parameter scales, respectively. Our motivation behind using these models were fourfold: (i) both BLOOM and GPT-3 show great capabilities in multiple languages, (ii) BLOOM models' usage is not as restricted via licensing (as opposed to other popular models such as LLaMA [165] or OPT [187]), (iii) GPT-3 has been one of the most popular

[17]https://huggingface.co/bigscience/bloom-1b7
[18]https://huggingface.co/bigscience/bloom-3b
[19]https://huggingface.co/bigscience/bloom-7b1

		Subtask 1			Subtask 2						
					BLOOM			GPT			
		GEN	HUM	Σ	1b7	3b	7b1	1b	6b7	175b	Σ
Spanish	Legal	4,846	4,358	9,204	640	665	712	919	942	919	4,797
	News	5,514	5,223	10,737	839	860	881	972	978	987	5,517
	Reviews	5,695	3,697	9,392	952	962	935	945	941	947	5,682
	Tweets	5,739	5,634	11,373	967	965	965	928	930	964	5,719
	How-to	5,690	5,795	11,485	894	929	960	970	983	966	5,702
	Total	27,484	24,707	52,191	4,292	4,381	4,453	4,734	4,774	4,783	27,417
English	Legal	5,124	5,244	10,368	809	779	832	890	887	927	5,124
	News	5,464	5,464	10,928	747	854	906	983	984	984	5,458
	Reviews	5,726	5,178	10,904	944	946	939	977	974	972	5,752
	Tweets	5,813	5,884	11,697	987	968	980	951	963	969	5,818
	How-to	5,862	5,918	11,780	962	976	982	993	993	963	5,869
	Total	27,989	27,688	55,677	4,449	4,523	4,639	4,794	4,801	4,815	28,021

Tab. 4.2.: AuTexTification 2023 dataset statistics presented by domain, class, and language in both subtasks.

and best performing language models until recently,[20] and (iv) we aimed to cover a broad spectra of model families and scales. While we were hoping to include BLOOM-175B generations too, this was not possible due to the lack of public APIs.

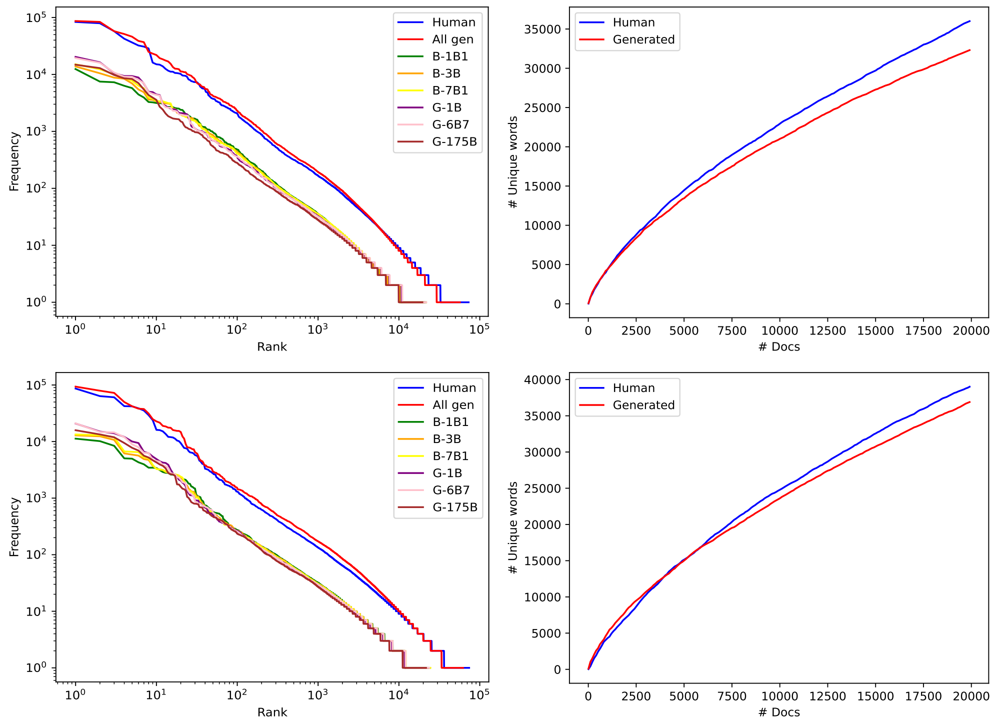
We manually tuned the decoding parameters to obtain MGT that appears realistic through subjective evaluations carried out by two of the authors. We found that with nucleus sampling [72], using a top-p of 0.9 and a temperature of 0.7, the models generated texts of higher quality. The maximum number of completion tokens was manually selected for each domain to be similar to the median token-length of the human texts: 20 tokens for tweets, 70 for reviews, and 100 for news, legal, and how-to articles.

The statistics of each subtask's data per domain, class, and language are presented in Table 4.2. In both subtasks, both languages contain similar amounts of texts, and the domains and classes are balanced in both splits. This way we guarantee that our analysis is fair by ensuring that every dimension is balanced. Furthermore, in Figure 4.2 we verify that the generated texts follow the Zipf (Figure 4.2a) and Heap's (Figure 4.2b) empirical laws, thus ensuring that the dataset is of high quality.

4.3.3 Evaluation of Generations

We first set out to verify the claims that in-domain MGT detection is an easy task by generating reviews in English and Spanish and training simple MGT detectors on this data. We also use this opportunity to experiment with various types of decoding strategies and models both in a subjective manner and using the objective evaluation metrics described in Section 2.5.

[20]GPT-3.5-turbo and GPT-4 were not released at time of compiling our dataset.



(a) Zipf plots.

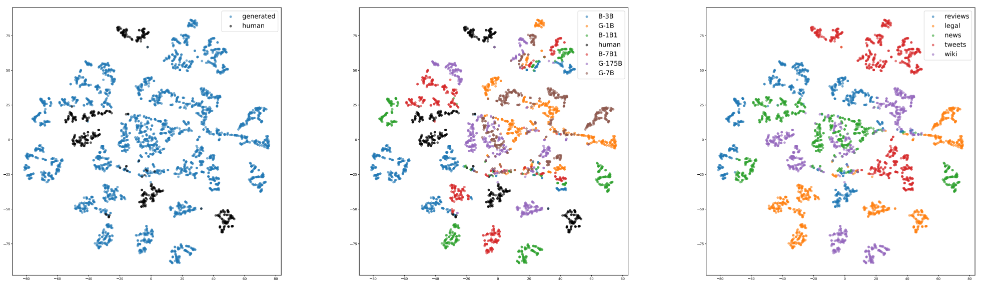
(b) Heaps plots.

Fig. 4.2.: Zipf and Heaps plots for English (top) and Spanish (below) for Subtask 1. B-prefix denotes BLOOM models and G- prefix denotes GPT models.

Our first efforts for generating text consisted in trying various smaller models: BLOOM-560m [147], GPT-2 [134], and XGLM-546M [98], with different decoding strategies for a single domain of reviews, in both English and Spanish. For this, we manually tried various parameters of each decoding strategy described in Section 2.4 to ensure the generated text was readable and of good quality, and ran various generations with these models. We then trained a simple logistic regression classifier with bag-of-word features, finding that the model achieves a Macro F_1 score of over 95% when detecting MGT.[21]

Having corroborated that in-domain MGT detection (and attribution) is easy, we evaluate the generations of the AuTexTification 2023 dataset using the metrics described in Section 2.5. Given the large amount of metrics, we first focus on metrics that measure the quality of generations (i.e. repetition, diversity, stop-word and symbol ratios, self-BLEU, perplexity), using these values as representation vectors of each text. We sample 500 texts per domain and author (i.e. human or each LLM), calculate these metrics, apply z-score normalization and use t-SNE [101] to project them[22] to a 2-dimensional space. The resulting plots are presented in Figure 4.3. In Figure 4.3a we present groupings by label, finding that there generally it is not

[21]We carry out a similar experiment with the same domain on the final data generated by larger models, finding similar results. Additionally, similar results were found for MGT attribution.
[22]We use a perplexity parameter of 15, keeping the remaining parameters as default.



(a) Grouped by label. (b) Grouped by model. (c) Grouped by domain.

Fig. 4.3.: t-SNE plots of generated text using evaluation metrics as representation vectors for both English and Spanish. Grouped by model. B- prefixes denote BLOOM models while G- prefixes denote GPT models.

difficult to distinguish, and that human text is more closely packed than generated text. This is interesting considering that MGT was generated using only six models, while the human data contains texts from thousands of authors. More over, in Figure 4.3b where the data is grouped by generation models, we observe that the data is very mixed between models, meaning that generally the attribution task will be difficult. Additionally, we find a cluster on the right that includes a mix of all the models. When observing domain-wise groupings of the data (Figure 4.3c), we find that this cluster corresponds to tweets data. This suggest that we should expect tweet data to be much more difficult to attribute to different models correctly. Additionally, the remaining domains are mixed with no clear distinctions between them, hinting that the detection and attribution in multi-domain settings is much more difficult than in an in-domain scenario. Similar plots for the previously-described first generation efforts are presented in Appendix A.

Furthermore, we evaluate the entailment of the generations with respect to the prompts used as inputs to the text generation models to ensure that the generated MGT maintains similar meaning to the input. We present the results for English grouped by domains and models in Tables 4.3 and 4.4, respectively. From these we find that most texts either have a positive or neutral entailment, but that certain amounts contradict the prompt. This is understandable and can be due to (i) the used NLI model not being good enough at capturing entailment in complex scenarios, or (ii) the text generation models being unable to always generate text that entails from the prompt. Specifically, from Table 4.3 we find that more MGT is contradicting the prompts in texts from the legal and wiki-how domains, showing that in more formal styles and related to specific niche areas, LLMs do not perform as well as in other areas. This is further observed in the tweets domain, where most MGT is entailed from the prompt, which also suggests that the NLI model may be biased shorter texts. Finally, Table 4.4 shows that the smaller GPT models, babbage and

Domain	Contradiction	Neutral	Entailment
Legal	994	1,995	2,135
News	434	2,818	2,212
Reviews	446	2,753	2,527
Tweets	214	939	4,660
WikiHow	654	2,755	2,453

Tab. 4.3.: Per domain entailments of prompts and generations.

Model	Contradiction	Neutral	Entailment
BLOOM-1B1	312	2,073	2,069
BLOOM-3B	319	1,963	2,205
BLOOM-7B1	363	2,012	2,264
babbage	766	1,719	2,309
curie	681	1,767	2,353
davinci	301	1,726	2,787

Tab. 4.4.: Per model entailments of prompts and generations.

curie, generate MGT that contradicts the prompts more commonly to other models. This could be a sign that these models are under-trained while the smaller BLOOM models are not.

We additionally evaluate the final generated data with its human counterparts using MAUVE [128], which was briefly described in Section 2.5. As previous analyses yielded similar results for both languages, we use MAUVE to evaluate only the English subset of the dataset. We present the MAUVE divergence curves of human and MGT texts in each domains, as well as of all the dataset in Figure 4.4. Similarly, MAUVE scores and the frontier integral between the human and MGT text distributions are presented in Table 4.5. From these, we observe that most of the domains' MGT is distributed similarly to the human text, but that the legal domain is especially different. However, after some manual inspection,[23] it was found that legal MGT was not very different from human-authored text. Thus we hypothesize that this low MAUVE score is due to the longer lengths of legal texts with respect to texts from other domains, and that MAUVE may be sensitive to texts of longer lengths. Unfortunately due to time and computational resource constraints we were unable to re-generate the legal texts and further explore this hypothesis. Generally we find similar measures to those obtained in other works exploring text generation [127].

[23]The authors of this work are not experts in the legal domain.

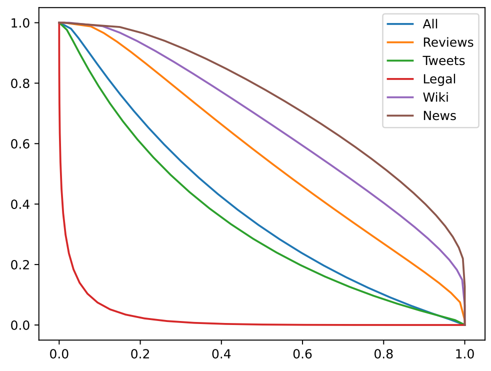


Fig. 4.4.: MAUVE divergence curves.

Domain	MAUVE	Frontier Integral
All	0.39	0.25
Reviews	0.57	0.17
Tweets	0.34	0.27
Legal	0.03	0.70
Wiki	0.66	0.14
News	0.74	0.12

Tab. 4.5.: MAUVE scores.

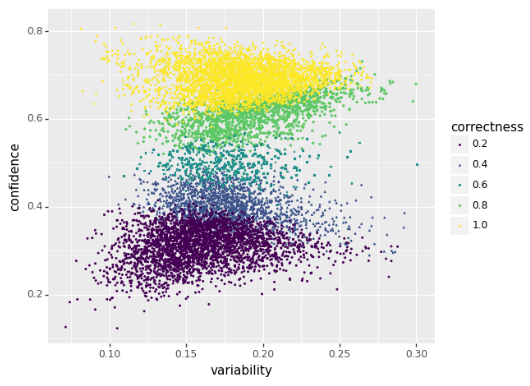
Finally, we present the results of a novel zero-shot dataset cartography technique [2] inspired by data maps derived from training dynamics [161], as a way to further understand the complexity of identifying MGT in real scenarios. These maps plot data-points by taking into account a model's confidence in the correct class and variability in the prediction through time. Brighter points correspond to examples that are easy-to-learn, with darker ones denoting hard-to-learn data-points, and points in the middle of the spectrum being ambiguous. With this technique, we get a better view of the complexity of the task and, specifically for our dataset, of the domain-wise distributions of texts. Figure 4.5 shows data maps grouped by domain for Subtask 1 in English. From here we further observe the differences between the legal domain with respect to the rest: there is much less variability in the confidence range between 0.4 and 0.6, which is not true for the other domains. However, the remaining domains are similarly distributed, with larger variability in the tails of confidence values. The same conclusions are found in the Spanish split (see Appendix A).

4.4 Human Assesment

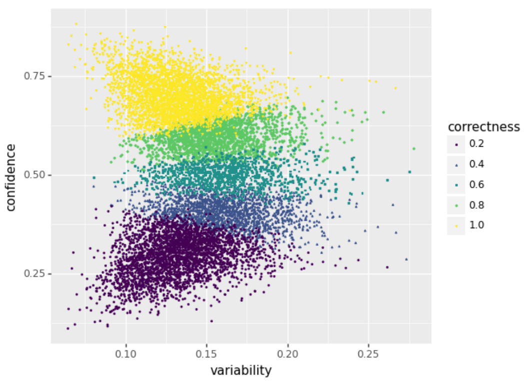
We performed a small-scale study to assess the difficulty of the Subtask 1 for human annotators. The study consisted in asking human annotators to classify texts as human or generated.[24] Five annotators were involved: four Spanish native speakers (SP) and one Italian native speaker (IT); all of them with C1-C2 proficiency level in English. From these annotators, SP1 and SP4 are familiar with generated text (they created the dataset and analysed hundreds of examples), while the others were exposed to the task for the first time.

We provided the same 40 texts to each annotator, drawn from the test set of the Subtask 1 both for English and Spanish. The texts were balanced in terms of classes

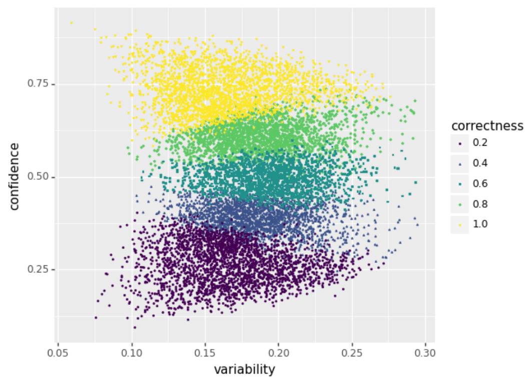
[24]The annotation interface and instructions are available at `https://colab.research.google.com/drive/1_zgW3k3DEV9Iv-wraH-DTqjm24pfT6lx`



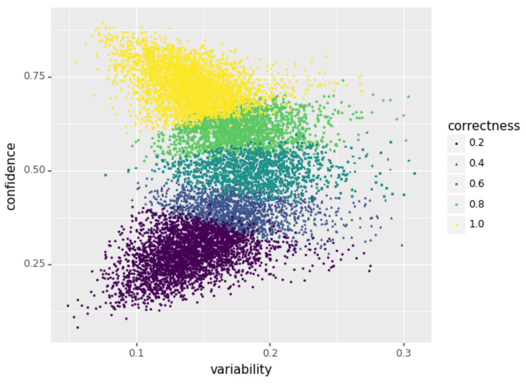
(a) Legal.



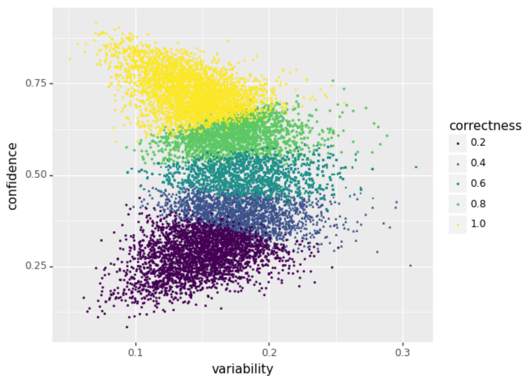
(b) News.



(c) Reviews.



(d) Wiki-How.



(e) Tweets.

Fig. 4.5.: Data maps for Subtask 1 in English grouped by domain.

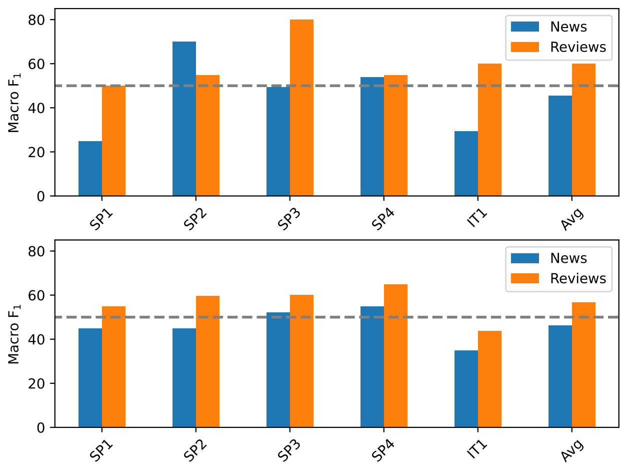


Fig. 4.6.: Human performance in English (top) and Spanish (bottom). The grey dotted line is the random baseline.

and domains: 20 texts were generated by LLMs and 20 were written by a human, half of them were news and the other half were reviews. The generated texts were only obtained from BLOOM models: 6 texts from BLOOM-1b, 6 texts from BLOOM-3b1, and 8 texts from BLOOM-7b1. Figure 4.6 shows the Macro-F$_1$ score of each annotator in each domain.

For both languages, the average annotator performance is very similar, most annotators are close to the random baseline. Regarding the domains, it seems more difficult for humans to distinguish between human-authored and machine-generated news rather than reviews. Most of the annotators perform worse than the random baseline distinguishing texts from the news domain. On the contrary, humans are typically better than the random baseline in the reviews domain, especially in English.

Language proficiency seems to play a role. IT1 shows better performance in English than in Spanish, where they are not proficient. Despite how SP1 and SP4 are familiar with generated texts, there is no significant difference between them and other annotators.

The human annotators did not follow any systematic pattern to detect MGT. For reviews, some mentioned that the generated reviews seemed generic, describing many general aspects with short sentences. In contrast, human reviews focused on few and more concrete aspects.

4.5 System's Overview

In this section, we briefly introduce the participants' systems, describe the baselines and evaluation metrics, and study the results of the shared task.

4.5.1 Submitted Approaches

The AuTexTification shared task received submissions from 36 teams, belonging to 30 different institutions and 18 different countries. All teams participated in the English track of Subtask 1, with 23 teams also taking part in the Spanish track. For Subtask 2, 19 teams participated in the English track and 14 in the Spanish track. Teams were allowed to submit a maximum of 3 runs per subtask and language. Overall, AuTexTification received a total of 175 runs, comprising 71 for the English track of Subtask 1, 47 for the Spanish track, 33 for the English track of Subtask 2, and 24 for the Spanish track. Outside of the competition scope, the AuTexTification datasets have been used in NLP courses within academic institutions. We are aware of at least 3 institutions,[25] with 17 participating teams and 58 runs.

Following the trend in the Natural Language Processing (NLP) field, most teams relied on pre-trained Transformer [169] models. The most used ones were BERT-based models [39] like RoBERTa [100] and DeBERTa [69]. Also, domain-specific and multilingual variants of BERT were frequent, including XLM-RoBERTa [27], RemBERT [25], and Twhin-BERT [189]. A smaller set of participants included generative models in their systems such as GPT-2 [134], Grover [186], and OPT [187].

Most of the best performing approaches used ensembles of pre-trained models, as well as combinations of lexical, stylometric or statistical features. In some cases, participants fine-tuned their models using auto-train procedures and performed hyper-parameter tuning. Some teams also included Convolutional Neural Networks [91] or Long Short Term Memory (LSTM) Networks [70] as part of their systems. Traditional machine learning models like Logistic Regression and Support Vector Machines (SVM) [29] were also frequent among the participants. However, these approaches generally performed worse than Transformer-based approaches.

There was also a great diversity in terms of features. Probabilistic token-level features from open-source generative language models seem to play an important role in the best performing approaches. Most of the participants employed contextual representations from pre-trained models, either as features, or through end-to-end

[25] Universitat Politècnica de València, Aix-Marseille Université, and IMT Atlantique.

finetuning. In addition to contextual representations, linguistic features including lexical, structural, and discourse features were also frequent. Among the most common linguistic features, we observed bag of word/char n-grams, counts of personal pronouns, stopwords, punctuations, and POS tags. Some participants also incorporated linguistic and factual knowledge directly in their models. Among these, we found the inclusion of syntactic dependencies in pre-trained models through contrastive learning, Wikipedia fact-checking, and native language identification.

The best ranked systems for each subtask ranged from complex ensembles of many different models and features, to single generative models fine-tuned for the task. In Subtask 1, both for English and Spanish, the best system was proposed by **TALN-UPF**. This system relied on a bidirectional LSTM [148] model trained with a combination of probabilistic token-level features from different GPT-2 versions, linguistic token-level features such as word-frequencies or grammar errors, and text representations from pre-trained encoders. Besides, **TALN-UPF** was the only team that considered a cross-domain evaluation in the validation step, by performing cross-validation over topically-split data after inferring the topics using Latent Dirichlet Allocation [15]. In the Spanish track, the **TALN-UPF** system performed similar to the **Lingüística_UCM** system, which was an SVM trained with a set of morphological, lexical, and discourse features selected according linguistic expertise and human analysis.

In Subtask 2, both for English and Spanish, the three runs of the **Drocks** team were the highest ranked ones. These systems were ensembles of five different Transformer-based classifiers fine-tuned on the task. The best ensembles differed for each language. For English, the best ensemble was an Error-Correcting Output Codes [40] model trained using the concatenation of the classification probabilities as features. For Spanish, the best ensemble was implemented with an SVM using the average of the classification probabilities as features.

4.5.2 Baselines

We consider several baselines for each subtask and language. Namely, we include a random baseline (Random), zero-shot (SB-ZS) and few-shot (SB-FS) approaches based on text and label embedding similarities, a bag-of-words encoding with logistic regression (BOW+LR), Low Dimensional Semantic Embeddings (LDSE), and fine-tuned language specific transformers (Transformer), DeBERTaV3 [69][26] for English and RoBERTa-BNE [50][27] for Spanish. These baselines consist in the following:

[26]`https://huggingface.co/microsoft/deberta-v3-base`
[27]`https://huggingface.co/PlanTL-GOB-ES/roberta-base-bne`

Random. The random baseline assuming class balance. Defined as $\frac{1}{C}$ where C is the number of classes.

SB-ZS and SB-FS. Zero-shot and Few-Shot Symanto Brain API,[28] a proprietary Symanto solution optimized for highly efficient and scalable state-of-the-art zero-shot and few-shot classification [110]. We verbalize labels for Subtask 1,[29] but not for Subtask 2 given the anonymity of the classes. For SB-FS we use 1024 shots.

BOW+LR. We encode the texts with bag of n-grams, using the top 5K word n-grams, $n \in \{1, 2\}$ and character n-grams, $n \in \{2, \ldots, 6\}$ following [129]. We train a Logistic Regression model offered by scikit-learn [123] with default parameters on z-score normalized and concatenated features.

LDSE. The Low Dimensionality Semantic Embedding (LDSE) model [136] represents texts on the basis of the probability distribution of occurrence of their tokens in the different classes. Using LDSE, we train an SVM classifier provided by scikit-learn [123] with default parameters.

Transformer. We use the HuggingFace ecosystem [180] to fine-tune a pre-trained Transformer with a randomly initialized classification head for 5 epochs and default hyperparameters. We use a batch size of 32 texts for DeBERTaV3 and a batch size of 64 for RoBERTa-BNE.

4.5.3 Evaluation

The submissions for both subtasks are evaluated with the Macro-F$_1$ score. Statistical significance is computed through bootstrapping with replacement at a confidence level of $\alpha = 0.95$ with 1,000 resamples. Additionally, we measure precision and recall scores, but do not consider these as official evaluation metrics for the tasks. We only use them to carry out fine-grained analysis of the submitted systems.

4.6 Subtask 1: MGT Detection

For the MGT detection subtask, we received 71 submissions from 36 different teams in English, and 47 submissions from 23 teams in Spanish. Tables 4.6 and 4.7 show the top-3 performing teams, the weakest team, as well as the first team that beats each baseline, both for English and Spanish.

[28] `https://www.symanto.com/nlp-tools/symanto-brain/`
[29] HUM: “*This text has been written by a human.*”
GEN: “*This text has been automatically generated by a bot.*”

Rank	Team	Run	Macro-F$_1$
1	TALN-UPF	HB_plus	**80.91**
2	TALN-UPF	HB	74.16
3	CIC-IPN-CsCog	run2	74.13
22	turquoise_titans	run1	65.79
23	BOW+LR	baseline	65.78
33	turing_testers	run3	60.64
34	LDSE	baseline	60.35
37	OD-21	run3	59.49
38	SB-FS	baseline	59.44
51	swissnlp_team	run2	57.20
52	Transformer	baseline	57.10
69	UMZ	run1	50.18
70	Random	baseline	50.00
74	SB-ZS	baseline	43.47
77	UAEMex	run1	33.87

Tab. 4.6.: Ranking of Subtask 1 in English.

Rank	Team	Run	Macro-F$_1$
1	TALN-UPF	HB_plus	**70.77**
2	Ling_UCM	run1	70.60
3	Transformer	baseline	68.52
20	GLPSI	run3	63.90
21	LDSE	baseline	63.58
25	turing_testers	run1	62.77
26	BOW+LR	baseline	62.40
39	bucharest	run2	56.49
40	SB-FS	baseline	56.05
46	ANLP	run1	51.38
47	Random	baseline	50.00
50	UAEMex	run3	35.17
51	SB-ZS	baseline	34.58
53	LKE_BUAP	run3	31.60

Tab. 4.7.: Ranking of Subtask 1 in Spanish.

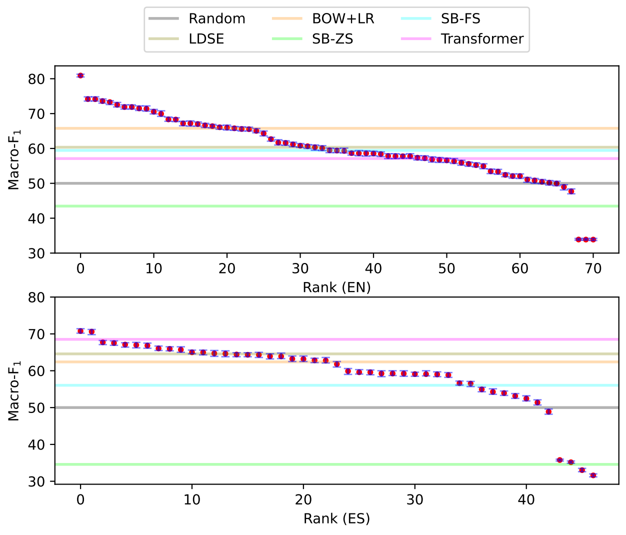
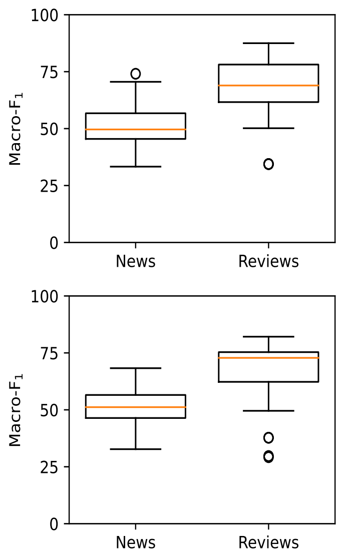
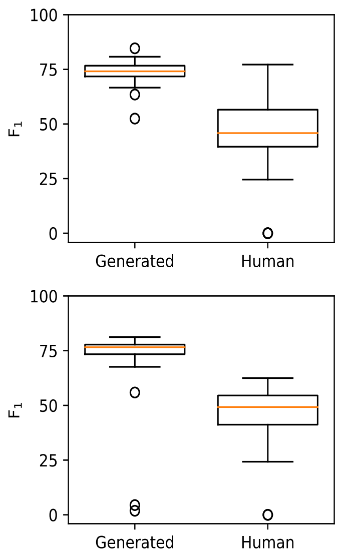


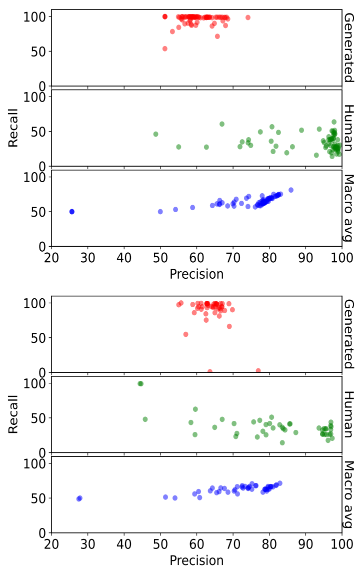
Fig. 4.7.: Rank-ordered Macro-F$_1$ with error bars for Subtask 1 in English (top) and Spanish (bottom). Colored lines are baselines.

The best system was proposed by the **TALN-UPF** team, with 80.91 and 70.77 Macro-F$_1$ scores in English and Spanish. In English, the best team is significantly better than the second-best ranked team. However, in Spanish there are no significant differences between the two best teams and the best baseline. In Figure 4.7, we illustrate the rank-ordered Macro-F$_1$ scores for all the teams in both languages.

Many teams surpassed the best baseline in English by large margins, whereas for Spanish only two teams were able to outperform it with small differences in Macro-F$_1$. Moreover, the performance of the top-11 ranked teams in English is higher than the performance of the best team in Spanish. This could suggest that detecting MGT and generalizing to new domains is easier in English than in Spanish, either due to language idiosyncrasies or because of the larger availability and quality of English NLP models. For both languages, we observe a linear relationship between







(a) Per domain Macro-F$_1$. **(b)** Per class F$_1$. **(c)** Per class precision-recall.

Fig. 4.8.: Fine-grained plots for Subtask 1 in English (top) and Spanish (bottom).

the the rank-ordered Macro-F$_1$ scores, with a small set of outliers in both tails. This hints that, even though the resulting Macro-F$_1$ scores in each language are in different ranges, there is similar variability and difficulty in both languages. The teams' systems cover almost the entire Macro-F$_1$ range in both languages, and, in many cases, they are very similar (same Transformer-based models, similar linguistic features, etc.). Therefore, one has to be careful when developing a MGT detector, small changes could lead to large improvements or declines.

We also include fine-grained results per-domain and per-class in Figure 4.8. When observing the domain-wise Macro-F$_1$ scores in Figure 4.8a, we find that the systems generalized better to reviews than to news, with a mean Macro-F$_1$ below the random baseline for the latter. Furthermore, both domains show long-tailed distributions, revealing the variability in generalization capabilities of the systems. Concerning class-wise F$_1$ scores in Figure 4.8b, we find that the systems are better at classifying generated text, and there is lower dispersion among the systems' F$_1$ scores for this class than for the human class. From the precision-recall distributions depicted in Figure 4.8c, we observe that systems are more biased towards predicting text to be generated (high recall), often doing so incorrectly (low precision). We observe the opposite for human texts, few predictions (low recall) but mostly correct (high precision). All the conclusions above hold for both languages.

In Table 4.8 we present the examples with the largest number of correctly and incorrectly predicted labels for Subtask 1. Here we see that two generated texts with most fails were rather short, while the incorrectly classified texts from HUM are longer and more intricate. Moreover, we find that it is easy for MGT detectors to

Text	Domain	Label	#Fails	#Correct
Lo unico que la habitacion del segundo piso tenia humos y olia mal	Reviews	Generated	44	3
La Comisión Europea (CE), que había amenazado con un procedimiento de infracción por ver falta de independencia en el proyecto normativo, vio en febrero con buenos ojos los cambios introducidos, pero la vicepresidenta de la CE, Neelie Kroes, advierte ahora de que algunas preocupaciones comunitarias en torno a la futura legislación no han sido abordadas y recalca que se adoptarán las medidas necesarias si los problemas persisten cuando entre en vigor.	News	Human	46	1
1. Nubank (Brasil): Nubank es una fintech brasileña que revolucionó el mercado financiero en Brasil al ofrecer una tarjeta de crédito sin anualidad. En 2018, Nubank superó los US$1.000 millones de valoración y se convirtió en el primer unicornio brasileño. 2. Rappi (Colombia): Rappi es una startup colombiana de entrega de productos y servicios. La compañía fue fundada en 2015 y desde entonces ha crecido rápidamente. En 2018, Rappi	News	Generated	0	47
Peor imposible, con deciros q después de 30 min, nos fuimos sin comer	Reviews	Human	2	45
you have to listen to him in the new mix..it shines like no other!!!!!	Reviews	Generated	59	12
Police Scotland said her death was being treated as unexplained pending further inquiries. But officers said there was nothing to suggest her death was suspicious and they were not looking for anyone else. Ms Kocisova is understood to have lived and worked in the Stirling area for a number of years.	News	Human	71	0
This is very durable. This is very easy to assemble. This is easy to install. This is a very good product. This is a durable product. This is a good product. This is a product that will help you sleep better at night. This is a good product. This is a great product. This is a product that is easy to install. This is a great product.	Reviews	Generated	0	71
Certainly applies here. Chamber maid cleaner is held together by thin plastic strips that break easily. Never really scraped all the coffee of the bottom anyway. Grind size selector is just a gimmick, much easier just to grind until you get the grind size you desire by holding down a bottom and taking a peak. Take my advice and go with a simple model.	Reviews	Human	3	68

Tab. 4.8.: Examples with largest number of correct and incorrect predictions along all systems in Subtask 1.

Rank	Team	Run	Macro-F$_1$
1	Drocks	run3	**62.50**
2	Drocks	run1	61.29
3	Drocks	run2	61.27
4	ViDa	run1	60.99
5	Transformer	baseline	60.42
31	LKE_BUAP	run1	45.62
32	LDSE	baseline	44.56
33	turquoise_titans	run2	43.37
34	BOW+LR	baseline	39.98
35	UAEMex	run2	33.19
36	SB-FS	baseline	28.94
37	Random	baseline	16.66
38	SB-ZS	baseline	15.70
39	ANLP	run1	14.61

Tab. 4.9.: Ranking of Subtask 2 in English.

Rank	Team	Run	Macro-F$_1$
1	Drocks	run2	**65.37**
2	Drocks	run3	64.72
3	Drocks	run1	64.17
7	TALN-UPF	Hybrid_plus	61.45
8	Transformer	baseline	61.34
20	iimasPLN	run1	51.43
21	LDSE	baseline	45.46
22	BOW+LR	baseline	45.31
25	UAEMex	run2	33.78
26	SB-FS	baseline	31.38
28	ANLP	run1	17.93
29	Random	baseline	16.66
30	SB-ZS	baseline	16.23

Tab. 4.10.: Ranking of Subtask 2 in Spanish.

correctly identify text to be generated when there is large amounts of structure repetitions, as can be observed in the penultimate row of the table. However, in the other correctly-predicted texts, we find it difficult to attest to why this is the case. Subjectively, most appear to be difficult to assign to either class with high certainty. Having found this observation, we carry out a more in-depth analysis of the texts and the predictions.

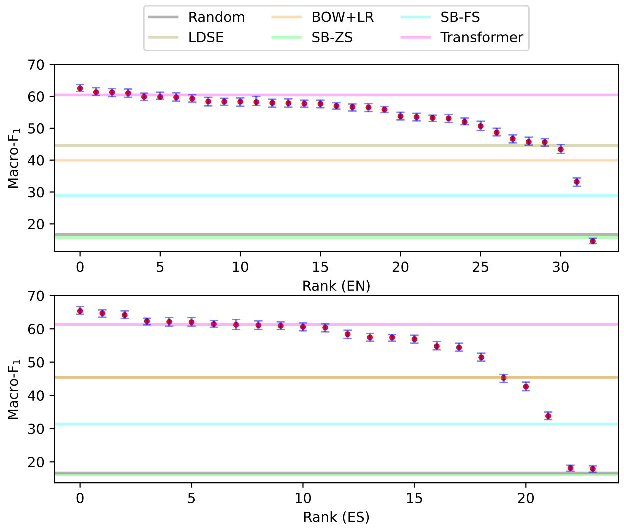
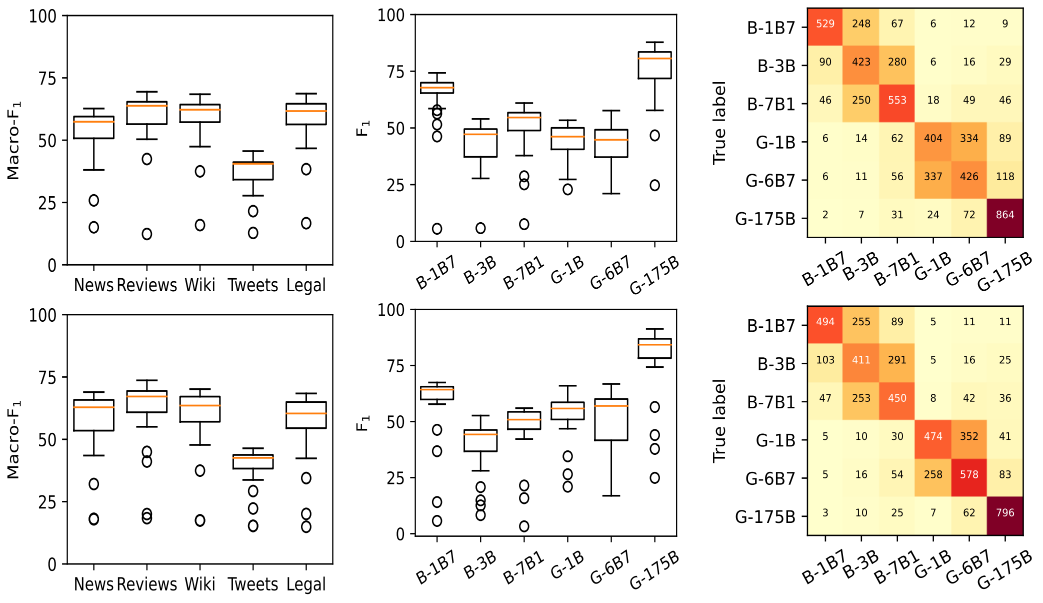


Fig. 4.9.: Rank-ordered Macro-F_1 for Subtask 2 in English (top) and Spanish (bottom). Colored lines are baselines.

4.7 Subtask 2: MGT Attribution

For the MGT Attribution subtask we received 33 submissions from 19 different teams in English, and 24 submissions from 14 teams in Spanish. Tables 4.9 and 4.10 show the top-3 performing teams, the weakest team, as well as the first team that beats each baseline, both for English and Spanish. The best system was submitted by team **Drocks**, obtaining 62.5 and 65.37 Macro-F_1 scores for English and Spanish, respectively. This is in contrast to the best scores of Subtask 1 nearing 80 and 70 Macro-F_1, showing that in-domain MGT attribution is more difficult than out-of-domain MGT detection. In this subtask, teams did not deviate significantly from the baselines, and for both languages the relative ranking of baselines remained the same, as opposed to Subtask 1. Rank ordered Macro-F_1 scores for both languages are presented in Figure 4.9. Few teams were able to surpass the best baselines, with most submissions performing between the top-2 baseline scores. Similarly to Subtask 1, we observe a linear relationship between rank and Macro-F_1 with outliers in the right tail, meaning that there is variability and difficulty in attributing MGT irrespective of language. However, teams cover a smaller range of Macro-F_1 scores, suggesting there is less variability when attributing MGT than detecting it, when compared to the variability in Subtask 1. In contrast to Subtask 1, teams generally obtained better Macro-F_1 scores in Spanish than English, but the differences were marginal when compared to Subtask 1, which could be because of the randomness of the learning procedures or due to a smaller number of participants in Subtask 2. Generally, MGT attribution appears promising but limited, suggesting the need for further research into new approaches or framings of the problem.



(a) Per domain Macro-F_1. (b) Per class F_1. (c) Median confusion matrices.

Fig. 4.10.: Fine-grained plots for Subtask 2 in English (top) and Spanish (bottom). B- prefix denotes BLOOM models and G- prefix denotes GPT models.

Fine-grained results for Subtask 2 per-domain and per-class are presented in 4.10. Per-domain results (Figure 4.10a) show that attribution of generated tweets is much more difficult that the remaining domains. For tweets, systems are unable to reach 50% Macro-F_1, while for the other domains they surpass it by a large margin. We additionally find many outliers toward lower scores, indicating the difficulty of the task. Finally, most domains have similar distributions centered around different medians, meaning that the variability of participating systems is maintained through all five domains. We also present per-class results in Figure 4.10b, where we find that it is easier to attribute generated text to BLOOM-1B1 and text-davinici-003. Moreover we observe large variability for curie, while the other classes have narrower distributions.

Additionally, we computed overall confusion matrices by taking the median at each position of the confusion matrix from all the participant's systems. Figure 4.10c shows the results for English and Spanish. In both languages, the largest confusions are across models within the same families, suggesting that it is easy to distinguish generation models of different families. Besides, text-davinci-003 is the model with less number of confusions, being different enough to be easily distinguished from the other models.

This chapter included contents from our AuTexTification 2023 Shared Task overview. For additional details, please refer to Sarvazyan et al. [145].

5 Generalization Capabilities of MGT Detectors

5.1 Introduction

In Section 4.5 we found that most teams participating in the AuTexTification Shared task submitted systems comprised of, or including, fine-tuned Transformer-based LMs for detecting and attributing MGT. While in this shared task we evaluated their capability to generalize to unseen domains, other forms of generalization were not studied. Given the speed at which new and better LLMs are released to the public, it is important for MGT detectors to be able to adapt to unseen LLMs of different types and parameter scales. In this section, we describe the methodology, datasets, experiments, and results involved in our analysis of the generalization capabilities of MGT detectors across families and scales of text generation models.

5.2 Experimental Set-up

We frame the experiments to study the generalization of MGT detectors as binary classification tasks. For cross-family generalization, we train detectors with human text and MGT from a single family, then we evaluate them on detecting human text and MGT from different families, one family at a time. Following the same methodology, we also study the MGT detectors' generalization across parameter scales.

We use the AuTexTification Shared Task datasets from both *Subtask 1: MGT detection* and *Subtask 2: Model Attribution*, leveraging it by fine-tuning Transformer-based MGT detectors and studying their capabilities We use the MGT and the labels from Subtask 2 by grouping them to obtain a training and test split per family. We add human text from Subtask 1 to these splits, matching the amount of MGT, thus obtaining our final training and test splits for each family. This way, we ensure that all the domains are in the training and test splits, and our data is balanced with respect to domains, classes (generated and human) and generators within families (or scales) in both splits. The same procedure is carried out to obtain per-scale training and test splits for scale-wise generalization. The final data statistics

Split	Family	**English**	**Spanish**
Train	**BLOOM**	10,897	10,511
	GPT	11,519	11,424
Test	**BLOOM**	2,714	2,615
	GPT	2,891	2,867

Tab. 5.1.: Family generalization data statistics.

Split	Scale	**English**	**Spanish**
Train	**1b**	7,432	7,210
	7b	7,509	7,345
	175b	3,827	3,866
Test	**1b**	1,811	1,816
	7b	1,931	1,882
	175b	988	917

Tab. 5.2.: Scale generalization data statistics.

of the family and scale generalization splits are presented in Tables 5.1 and 5.2, respectively.

For training our classifiers at each experiment, we employ three models: a language specific model (DeBERTa [69] for English, and MarIA [49] for Spanish) a multilingual model (XLM-RoBERTa [27]), and a small model from a family of generators used to compile the AuTexTification dataset (BLOOM-560M [147]).[30] We fine-tune these models, with a randomly initialized classification head, in FP16 for 5 epochs using a linearly decaying learning rate schedule starting at 5e-5. Finally, we evaluate the models using class-wise and macro F_1 scores. All the experiments have been conducted using the HuggingFace ecosystem [181].

5.3 Generalizing to Unseen Model Families

To study cross-family generalization, we split the generated text into two groups: GPT and BLOOM. We train MGT detectors with human-authored text and text from one family, then evaluating on both families separately. The results are presented in Tables 5.3 and 5.4 for English and Spanish, respectively.

In both languages we observe how all MGT detectors perform much better when tested on the same family, reaching differences of 29 macro F_1 with respect to

[30]For the sake of fairness, our generalization experiments exclude the BLOOM models originally used to create the datasets. Likewise, we exclude GPT models given their limited transparency in the offered fine-tuning methodologies which could lead to unfair comparisons against the chosen classifiers.

		BLOOM			GPT		
Train	Detector	GEN	HUM	Mean	GEN	HUM	Mean
BLOOM	BLOOM-560	93.70	93.92	93.81	59.32	75.81	67.57
	DeBERTa	**95.21**	**94.79**	**95.00**	76.19	80.66	78.43
	XLM-R	93.13	92.14	92.63	**79.26**	**80.86**	**80.06**
GPT	BLOOM-560	72.17	79.82	75.99	89.61	89.78	89.69
	DeBERTa	**85.61**	**85.05**	**85.33**	**89.94**	**87.82**	**88.88**
	XLM-R	82.40	82.04	82.22	89.52	87.22	88.37

Tab. 5.3.: F_1 scores of the detectors for the generated (GEN) and human (HUM) classes when trained and evaluated on BLOOM and GPT model families (English). Best results in bold.

cross-family evaluation when training with BLOOM in Spanish. Overall, **detectors do not generalize well to other families**.

In English, the language-specific detector (DeBERTa) outperforms the multilingual detectors in most scenarios. This also holds in Spanish for in-family evaluation, whereas in a cross-family setting the language-specific model, MarIA, lags behind XLM-R, with BLOOM-560 again having the worst performance, meaning that **language-specific detectors are generally preferable**.

In both languages BLOOM-560 obtains lower F_1 scores in the generated class than DeBERTa and XLM-R when trained with GPT and evaluated with BLOOM. Differences in terms of F_1 scores regarding the other detectors are generally large, with the largest difference being of 13 points in English and 18 in Spanish. More research is needed to determine whether family-specific detectors generalize well to their own families. Nonetheless, from this experiment we conclude that **BLOOM-560 does not generalize well to its family**.

In cross-family settings, and independently of the language, **most detectors obtain higher F_1 scores on the human class than in the generated one**, reaching 22 points of difference. This may be because the generated class contains MGT of different quality levels from the same family, while human text quality is consistently similar.

Finally, **the training family of generators matters**: cross-family generalization depends on the training family of an MGT detector. For example, when training with MGT from BLOOM and evaluating on GPT in English, all detectors obtain worse results than in the opposite generalization direction. Interestingly, this behaviour is reversed in Spanish, where detectors trained with BLOOM and evaluated on GPT perform better. Thus, **one must carefully choose the training families** when building datasets to train supervised detectors in order to generalize well to other model families. Besides, **this choice may be different for different languages**.

		BLOOM			GPT		
Train	Detector	GEN	HUM	Mean	GEN	HUM	mean
BLOOM	BLOOM-560	88.05	87.78	87.91	65.03	73.52	69.28
	MarIA	**96.25**	**96.29**	**96.27**	58.95	75.91	67.43
	XLM-R	91.74	90.32	91.03	**73.93**	**76.29**	**75.11**
GPT	BLOOM-560	52.68	73.91	63.30	90.69	91.12	90.91
	MarIA	56.91	75.64	66.27	**94.97**	**94.98**	**94.98**
	XLM-R	**70.58**	**76.76**	**73.67**	91.14	89.50	90.32

Tab. 5.4.: F_1 scores of the detectors for the generated (GEN) and human (HUM) classes when trained and evaluated on BLOOM and GPT model families (Spanish). Best results in bold.

		1b			7b			175b		
Train	Detector	GEN	HUM	Mean	GEN	HUM	Mean	GEN	HUM	Mean
1b	BLOOM-560	89.69	90.04	89.89	85.22	86.45	85.84	76.37	83.43	79.90
	DeBERTa	**93.46**	**92.88**	**93.17**	**91.84**	**91.04**	**91.44**	89.90	**91.45**	90.67
	XLM-R	89.29	86.96	88.13	87.87	84.67	86.27	**91.12**	90.86	**90.99**
7b	BLOOM-560	87.49	**88.25**	**87.87**	86.02	**86.72**	**86.37**	79.16	84.75	81.96
	DeBERTa	**88.71**	85.99	87.35	**87.20**	83.14	85.17	**92.38**	**92.03**	**92.20**
	XLM-R	86.92	82.89	84.91	85.30	79.59	82.45	90.02	88.87	89.44
175b	BLOOM-560	56.14	74.47	65.30	64.47	77.36	70.92	91.52	**91.97**	91.75
	DeBERTa	69.77	75.51	72.64	**81.36**	**81.86**	**81.61**	**92.64**	91.48	**92.06**
	XLM-R	**73.31**	**75.67**	**74.49**	**81.36**	80.61	80.99	90.50	88.45	89.48

Tab. 5.5.: F_1 scores of the detectors for the generated (GEN) and human (HUM) classes when trained and evaluated on 1b, 7b and 175 parameter scales (English). Best results in bold.

5.4 Generalizing to Unseen Parameter Scales

Similarly to the cross-family experiment, we train MGT detectors with human-authored text and text from one parameter scale of models. In this case, given the selection of models used to compile the AuTexTification datasets, we opt for three groups: **1b**, comprised of BLOOM-1b7 and babbage; **7b**, consisting of BLOOM-7b1 and curie; and **175b** which only includes text-davinci-003. This last group is only comprised of GPT models given their popularity and the lack of APIs that provide access to BLOOM-175b or other LLMs with similar parameter scales. We carry out in-scale and cross-scale evaluation in English (Table 5.5) and Spanish (Table 5.6).

We observe that most cross-scale evaluations result in +80 macro F_1, meaning that in general, **MGT detectors generalize well to other scales**, sometimes performing better than their in-scale counterparts as is the case of DeBERTa when evaluated in the 7b scale after training with MGT from 1b-scaled models. However, in some particular cases, we find **bad generalization from very large scales to small ones**. For example, when training on MGT from the 175b scale, the cross-scale performance is lower than in other scenarios, which can be due to this scale only including MGT generated by GPT and not of the largest BLOOM model. Interestingly, when we

		1b			7b			175b		
Train	Detector	GEN	HUM	Mean	GEN	HUM	Mean	GEN	HUM	Mean
1b	BLOOM-560	90.57	90.09	90.33	86.76	86.77	86.72	86.58	88.98	87.78
	MarIA	**94.13**	**94.25**	**94.19**	**90.90**	**91.54**	**91.22**	83.33	87.50	85.42
	XLM-R	87.85	84.35	86.10	86.67	82.62	84.64	**91.58**	**91.18**	**91.38**
7b	BLOOM-560	88.03	88.35	88.19	87.54	87.75	87.65	88.48	90.41	89.44
	MarIA	**91.75**	**92.00**	**91.88**	**92.52**	**92.54**	**92.53**	**93.43**	**94.20**	**93.82**
	XLM-R	85.61	80.24	82.92	84.64	78.37	81.51	90.16	88.69	89.43
175b	BLOOM-560	51.85	73.16	62.50	55.37	74.22	64.80	93.27	93.64	93.45
	MarIA	53.77	74.23	64.00	64.16	77.27	70.71	**96.29**	**96.30**	**96.29**
	XLM-R	**73.45**	**75.17**	**74.31**	**79.97**	**78.88**	**79.42**	90.74	88.80	89.77

Tab. 5.6.: F_1 scores of the detectors for the generated (GEN) and human (HUM) classes when trained and evaluated on 1b, 7b and 175 parameter scales (Spanish). Best results in bold.

		1b		7b		175b	
Train	True Labels	GEN	HUM	GEN	HUM	GEN	HUM
175b	GEN	75.16	78.31	75.05	83.27	71.75	78.24
	HUM	77.90	77.96	78.06	76.81	78.99	77.25

Tab. 5.7.: Mean Flesch Reading Ease Scores [51] for the predictions of XLM-R trained using the 175b data in English.

analyze this behaviour from the text readability and complexity viewpoint (see Table 5.7), we observe that the readability of generated texts incorrectly classified as human is generally similar to that of correctly classified human texts: they are both easier to read. This is also in line with the training instances, where the generated texts have a mean readability score of 72.13 in contrast to a 77.09 in human-authored texts. In addition, the average number of *difficult words*[31] is greater in the human class. This signal is captured by some models: when testing in the 1b and 175b scales, there are in average over 5% more difficult words in the predicted human class. Note that, when evaluating in the 175b scale in a cross-scale scenario, the detectors obtain reasonably good scores. In fact, it is possible to detect texts generated by text-davinci-003 with +90 F_1 using a training set comprised of text generated by models of 1b parameters. As in our previous study, this shows that one must **carefully choose the LLMs' scale** when building datasets to train supervised detectors in order to generalize well to other model scales.

Similarly to the cross-family experiment, we observe that **language-specific detectors perform better than multilingual ones**. However, in contrast to the previous experiment, for cross-scale generalization we find that models are typically **not biased towards the human class** given that the F_1 scores for each class are similar in most cases; in fact the opposite is sometimes true, especially when training with MGT from the 7b scale.

[31]Difficult words according to: `https://github.com/textstat/textstat`

Finally, we find that **BLOOM-560 does not generalize well when trained with the 175b scale** in both English and Spanish, obtaining macro F_1 scores of as much as 11 points lower than the best detector when evaluated on MGT from the 7b scale. This could be due to BLOOM-560 being trained with MGT that is very different to the distribution it had originally learned. Additionally, in English it obtains low results when generalizing to the 175b scale.

In this work we detailed the findings of our analysis in the capabilities of generalization of MGT detectors to new model families and parameter scales. For more information, we refer the reader to Sarvazyan et al. [146].

6 Attribution to Model Families and Parameter Scales

6.1 Introduction

We have already motivated the need for model attribution in Section 1.1. However, from the results of Subtask 2 in the AuTexTification Shared Task described in Section 4.7, we observed that attributing to each particular model can be a difficult problem. This might be due to the vast space of LLM types and parameters scales, and as more and more models are included in the label space, the task becomes much harder. In this section, we frame the problem of MGT attribution as a multi-label classification problem where models are trained to predict the model family and parameter scale of a generator instead of attributing MGT to a specific generator. We study the feasibility of this approach for attributing MGT to families and scales as a proxy to the original task, reducing the dimensionality of the output space and thus making the task simpler for MGT attributors to solve. This way, we separately identify both the family and the scale of the MGT generator.

6.2 Experimental Set-up

We approach model family and parameter scale attribution as classification tasks between families or scales. We exclude human-authored text to consider the scenario where attribution is applied after a text has been identified as MGT. We employ the same Transformer-based language models used for the generalization experiments described in Chapter 5, fine-tuning them with the same parameters on the following data.

To obtain the data for the experiments used to study the feasibility of attributing to model families and parameter scales, we employ texts from the AuTexTification 2023 dataset (see Section 4.3.2). Specifically, given that this is an MGT attribution task, we assume that the texts have been previously identified as MGT, and thus only make use of data from Subtask 2, grouping its content by families or scales. Moreover, given the lack of counterpart models of BLOOM-3B and text-davinci-003 in the other family, we exclude these models from the scale attribution experiments

	Train		Test	
	GPT	**BLOOM**	**GPT**	**BLOOM**
English	11,519	10,897	2,891	2,714
Spanish	11,424	10,511	2,867	2,615

Tab. 6.1.: Family attribution data statistics.

	Train		Test	
	1b	**7b**	**1b**	**7b**
English	7509	7432	1931	1811
Spanish	7345	7210	1882	1816

Tab. 6.2.: Scale attribution data statistics.

	English				Spanish		
Attributor	**BLOOM**	**GPT**	Mean	Attributor	**BLOOM**	**GPT**	Mean
BLOOM-560	90.55	91.23	90.89	BLOOM-560	91.25	92.46	91.86
DeBERTa	**94.09**	**94.51**	**94.30**	MarIA	94.77	95.25	95.01
XLM-R	93.97	93.97	93.97	XLM-R	**95.10**	**95.48**	**95.29**

Tab. 6.3.: F_1 scores of attributors of MGT in English and Spanish in the BLOOM or GPT families. Best results in bold.

to ensure the per-class data is not biased towards particular families. The final dataset statistics are presented in Tables 6.1 and 6.2 for attributing to families and scales, respectively.

6.3 Attributing to Model Families

We study the family attribution problem in English and Spanish by fine-tuning Transformer-based language models to classify MGT into two classes, BLOOM and GPT. The results are presented in Table 6.3, where we observe that attributors slightly favor GPT texts, obtaining better F_1 scores compared to the BLOOM class. Additionally, BLOOM-560 **does not perform on par with other attributors**, especially when attributing text to its own family. It does not find a bias towards MGT of its own family for attribution, which follows from what was observed in previous experiments. Moreover, **language-specific attributors are not necessarily better**, seeing as XLM-R performs on par with DeBERTa and MarIA. Given the observed +90 macro F_1 scores, we conclude that **MGT can be feasibly attributed to model families**, thus reducing the space of possible outcomes.

	English				Spanish		
Attributor	**1b**	**7b**	Mean	Attributor	**1b**	**7b**	Mean
BLOOM-560	56.47	60.59	58.53	BLOOM-560	59.90	57.56	58.73
DeBERTa	**67.15**	**69.93**	**68.54**	MarIA	**70.42**	**72.40**	**71.41**
XLM-R	65.23	0.00	32.61	XLM-R	65.87	0.00	32.93

Tab. 6.4.: F_1 scores of attributors of MGT in English and Spanish in the 1b or 7b scales. Best results in bold.

6.4 Attributing to Parameters Scales

We study cross-scale generalization in MGT attributors in English and Spanish by fine-tuning attributors to classify MGT into two classes: 1b, comprised of MGT from BLOOM-1b7 and babbage, and 7b which contains MGT generated by curie and BLOOM-7b1. We exclude BLOOM-3b and GPT davinci since, in our experiment data, they cannot be paired with other models from the other family in their respective parameter scales. The results are shown in Table 6.4, where we observe lower scores in comparison to the family attribution experiment.In this experiment, XLM-R always predict the 1b scale. We hypothesize that this could be either due to overfitting or some aspect that degenerates the training dynamics, such as the random seed or learning rate scheduler. The best attributor is the language-specific MarIA, which reaches 71 macro F_1. MarIA obtains a similar F_1 score both for both 1b and 7b scales, suggesting that either (i) generators of 1b and 7b scales generate text of similar quality, or (ii) they include MGT from different model families, meaning that within each scale class the texts do not have many underlying similarities. Thus, we conclude that, while not being as feasible as family attribution, **scale attribution is promising and has potential for high performance with further developments**.

This chapter included contents from our paper where we analyse the feasibility of MGT attribution in model families and parameter scales. For additional details, please refer to Sarvazyan et al. [146].

7 Conclusions and Future Work

This work studies the tasks of detecting and attributing machine-generated text, exploring open problems such as (i) MGT detection in realistic multi-domain scenarios, (ii) MGT attribution, and (iii) generalization of MGT detectors and attributors to new text generation models.

The main goals of this work, presented in Section 1.3, were successfully reached. Specifically, we compiled a multilingual, multi-domain, multi-style and multi-generator dataset of human and generated text and used it to organize a shared task in the fields of MGT detection and attribution in realistic scenarios (see detailed conclusions in Section 7.1.1), we studied the generalization capabilities of MGT detectors to new domains (see Section 7.1.1), as well as new model families and parameter scales (see Section 7.1.2), and explored the feasibility of attributing MGT to groups of text generation models (see Section 7.1.3). Importantly, the work derived from this research resulted in two scientific papers, one invited seminar at a scientific workshop, one organized shared task and one dataset (see Section 7.2).

In Chapter 1 we motivated the MGT detection and attribution tasks. In Chapters 2 and 3 we provided an overview of techniques for MGT generation and detection of MGT. Afterwards, we illustrated the AuTexTification 2023 dataset and shared task in Chapter 4, carrying out further experiments and analysis into the generalization capabilities of MGT detectors in Chapter 5 while also exploring the feasibility of attributing MGT to groups of text generation models in Chapter 6.

What follows is the conclusions of this work, where we detail how its objectives were accomplished, the scientific contributions derived from this work, and promising research lines to be explored in future endeavors.

7.1 Conclusions

7.1.1 AuTexTification 2023

In this work we compiled the AuTexTification 2023 dataset, a multilingual, multi-domain, multi-style and multi-generator dataset of MGT and human authored text with various annotations, with the purpose of evaluating MGT detection systems in a realistic setting. We compiled the dataset such that the MGT and human text can be fairly compared by both being continuations of the same prefix, and carried out extensive evaluations of the quality of the generations by themselves, with respect to the prompts and the human text distribution. This dataset was compiled successfully, having already been used in various NLP courses in important academic institutions.

Moreover, we leveraged this dataset to organize the AuTexTification 2023 Shared Task at IberLEF2023, which included two subtasks, (i) MGT detection in unseen domains and (ii) MGT attribution. We extensively studied the submitted approaches and the results, concluding that MGT detectors' generalization capabilities to new domains is promising, while multi-domain MGT attribution appears limited and needs further research. With a total of 114 registered teams, 36 participating teams, 175 submissions, and 20 teams that sent working notes, we consider the AuTexTification 2023 Shared Task a success in promoting research in the fields of MGT detection and attribution.

7.1.2 Generalization Capabilities of MGT Detectors to Model Families and Parameter Scales

Having observed that most submissions in the AuTexTification 2023 Shared Task included Transformer-based language models, we found it important to study their generalization capabilities to unseen text generation models of different model families and parameter scales. Specifically, we observed that (i) MGT detectors do not generalize well to other families but do generalize well to unseen scales, (ii) that language specific detectors should be preferred to multilingual ones, that (iii) the training family or scale is important and might differ between languages, and that (iv) detectors based on the same family as that of the training data typically perform worse.

7.1.3 Feasibility of Model Family and Parameter Scale Attribution

From the AuTexTification 2023 Shared Task we also found that MGT attribution is generally limited in the way that is commonly framed. Instead, we propose a new perspective where we separately attribute MGT to families and scales. With this approach, we find that family attribution is feasible and easy, with all MGT attributors consistently reaching Macro F_1 scores above 90%. However, attributing

MGT to scales appears to be the crux of the problem as most scale attributors are unable to reach scores higher than 73% Macro F_1.

7.2 Scientific Contributions

The various contributions of this work have been materialized in the several scientific publications. As a result, one conference and one journal paper have been generated, and one invited talk has been carried out in a scientific workshop. Below, we sum up the scientific contributions.

The creation of the AuTexTification 2023 dataset and the organization of the AuTexTification 2023 Shared Task resulted in the following journal paper:

- Sarvazyan, A. M., González, J., Franco-Salvador, M., Rangel, F., Chulvi, B., & Rosso, P. (2023). *Overview of AuTexTification at IberLEF 2023: Detection and Attribution of Machine-Generated Text in Multiple Domains*. In Procesamiento del Lenguaje Natural, 71.

The experiments carried out to study model family and parameter scale generalization of MGT detectors, as well as the feasibility of MGT attribution to model families and parameter scales, resulted in the following proceedings paper:

- Sarvazyan, A. M., González, J., Rosso, P., & Franco-Salvador, M. (2023). *Supervised Machine-Generated Text Detectors: Family and Scale Matters*. (accepted) In Proceedings of the Fourteenth Conference and Labs of the Evaluation Forum.

Moreover, the aforementioned research contributions have been presented in the following seminar:

- Sarvazyan, A. M. *AuTexTification 2023 and more: Detection and Attribution of Machine-Generated Text in Multiple Domains*. humain OU PaS humain (hOUPSh) at Conférence en Recherche d'Information et Applications, Conférence sur le Traitement Automatique des Langues Naturelles (CORIA-TALN). June 5, 2023.

7.3 Future Work

Various problems from the field of MGT detection and attribution, and their solutions, have been studied in this work. While immediate future work consists in presenting the scientific papers resulting from this work in their respective conferences, there is still large amounts of open and important problems to tackle in the field.

Future work can focus on carrying out a more extensive analysis of MGT corpora and proposing more complex models that leverage specific information that identifies MGT from human-authored text. An important direction for future work is to delve deeper into the distinctive features that identify MGT from human-authored text. This involves conducting detailed analyses to identify specific linguistic, semantic, or syntactic patterns that are characteristic of MGT. We hope to leverage these features and the gained insight to propose models that more accurately identify MGT.

Another important future endeavor consists in enhancing MGT detection generalization to unseen domains, writing styles, and languages. Similarly, given the speed at which LLMs are evolving and new models are being released, ensuring the robustness of MGT detectors and attributors with respect to these advances is of high importance, which may need the inclusion of techniques that allow for fast adaptation of MGT detectors to new LLMs.

An important open problem in MGT detection and attribution is their robustness towards adversarial attacks. Many types of adversarial techniques exist and can be easily used to deceive these models, which is why future work should also involve adversarial training strategies, countermeasures to known techniques.

Finally, we plan to train and deploy MGT detectors and attributors in real-world scenarios. This involves ethical considerations such as to whom the false positives may affect, as well as privacy concerns, meaning that the developed systems should ideally be transparent and interpretable.

Bibliography

[1] Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01. 2019, pp. 3159–3166 (cit. on p. 14).

[2] Anonymous authors. Zero-Shot Data Maps. Efficient Dataset Cartograpy Without Model Training. In: *Publication under review*. 2023 (cit. on p. 33).

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. In: *arXiv preprint arXiv:1607.06450* (2016) (cit. on p. 11).

[4] Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In: *3rd International Conference on Learning Representations, ICLR 2015*. 2015 (cit. on p. 9).

[5] Anton Bakhtin, Sam Gross, Myle Ott, et al. Real or fake? learning to discriminate machine from human generated text. In: *arXiv preprint arXiv:1906.03351* (2019) (cit. on pp. 21, 23).

[6] Francesco Barbieri, Luis Espinosa Anke, and Jose Camacho-Collados. XLM-T: Multilingual Language Models in Twitter for Sentiment Analysis and Beyond. In: *Proceedings of the Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, June 2022, pp. 258–266 (cit. on p. 28).

[7] Alberto Barrón-Cedeño, Giovanni Da San Martino, Mirko Degli Esposti, et al. Experimental IR Meets Multilinguality, Multimodality, and Interaction: 13th International Conference of the CLEF Association, CLEF 2022, Bologna, Italy, September 5–8, 2022, Proceedings. Vol. 13390. Springer Nature, 2022 (cit. on p. 25).

[8] Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state Markov chains. In: *The annals of mathematical statistics* 37.6 (1966), pp. 1554–1563 (cit. on p. 7).

[9] Nora Belrose, Zach Furman, Logan Smith, et al. Eliciting latent predictions from transformers with the tuned lens. In: *arXiv preprint arXiv:2303.08112* (2023) (cit. on p. 13).

[10] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. In: *Advances in neural information processing systems* 13 (2000) (cit. on p. 8).

[11] Misha Benjamin, Paul Gagnon, Negar Rostamzadeh, et al. Towards standardization of data licenses: The montreal data license. In: *arXiv preprint arXiv:1903.12262* (2019) (cit. on p. 2).

[12] Luca Bertuzzi. Ai Act: Meps want fundamental rights assessments, obligations for high-risk users. Jan. 2023 (cit. on p. 23).

[13] Stella Biderman, Hailey Schoelkopf, Quentin Anthony, et al. Pythia: A suite for analyzing large language models across training and scaling. In: *arXiv preprint arXiv:2304.01373* (2023) (cit. on p. 14).

[14] Sidney Black, Stella Biderman, Eric Hallahan, et al. GPT-NeoX-20B: An Open-Source Autoregressive Language Model. In: *Proceedings of BigScience Episode# 5–Workshop on Challenges & Perspectives in Creating Large Language Models*. 2022, pp. 95–136 (cit. on p. 14).

[15] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022 (cit. on p. 37).

[16] Dasha Bogdanova and Angeliki Lazaridou. Cross-Language Authorship Attribution. In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland: European Language Resources Association (ELRA), May 2014, pp. 2015–2020 (cit. on p. 23).

[17] Tom Brown, Benjamin Mann, Nick Ryder, et al. Language models are few-shot learners. In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901 (cit. on pp. 10, 14, 28).

[18] Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 11079–11091 (cit. on p. 9).

[19] Ilias Chalkidis, Manos Fergadiotis, and Ion Androutsopoulos. MultiEURLEX – A multi-lingual and multi-label legal document classification dataset for zero-shot cross-lingual transfer. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021 (cit. on p. 28).

[20] Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code. In: *arXiv preprint arXiv:2107.03374* (2021) (cit. on p. 1).

[21] Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In: *Computer Speech & Language* 13.4 (1999), pp. 359–394 (cit. on p. 7).

[22] Kyunghyun Cho, Bart Merrienboer, Caglar Gulcehre, et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In: *EMNLP*. 2014 (cit. on p. 9).

[23] Noam Chomsky. Three models for the description of language. In: *IRE Trans. Inf. Theory* 2 (1956), pp. 113–124 (cit. on p. 7).

[24] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, et al. Palm: Scaling language modeling with pathways. In: *arXiv preprint arXiv:2204.02311* (2022) (cit. on pp. 14, 15).

[25] Hyung Won Chung, Thibault Fevry, Henry Tsai, Melvin Johnson, and Sebastian Ruder. Rethinking Embedding Coupling in Pre-trained Language Models. In: *International Conference on Learning Representations*. 2020 (cit. on p. 36).

[26] Hyung Won Chung, Le Hou, Shayne Longpre, et al. Scaling instruction-finetuned language models. In: *arXiv preprint arXiv:2210.11416* (2022) (cit. on p. 10).

[27] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, et al. Unsupervised Cross-lingual Representation Learning at Scale. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020, pp. 8440–8451 (cit. on pp. 36, 46).

[28] Danish Contractor, Daniel McDuff, Julia Katherine Haines, et al. Behavioral use licensing for responsible AI. In: *2022 ACM Conference on Fairness, Accountability, and Transparency*. 2022, pp. 778–788 (cit. on p. 2).

[29] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In: *Machine learning* 20 (1995), pp. 273–297 (cit. on p. 36).

[30] Evan Crothers, Nathalie Japkowicz, Herna Viktor, and Paula Branco. Adversarial Robustness of Neural-Statistical Features in Detection of Generative Transformers. In: *2022 International Joint Conference on Neural Networks (IJCNN)*. 2022, pp. 1–8 (cit. on p. 22).

[31] Evan Crothers, Nathalie Japkowicz, and Herna L. Viktor. Machine Generated Text: A Comprehensive Survey of Threat Models and Detection Methods. In: *ArXiv* abs/2210.07321 (2022) (cit. on pp. 22, 23).

[32] Walter Daelemans, Mike Kestemont, Enrique Manjavacas, et al. Overview of PAN 2019: bots and gender profiling, celebrity profiling, cross-domain authorship attribution and style change detection. In: *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 10th International Conference of the CLEF Association, CLEF 2019, Lugano, Switzerland, September 9–12, 2019, Proceedings 10*. Springer. 2019, pp. 402–416 (cit. on p. 1).

[33] Zihang Dai, Zhilin Yang, Yiming Yang, et al. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019, pp. 2978–2988 (cit. on p. 15).

[34] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. In: *Advances in Neural Information Processing Systems*. 2022 (cit. on p. 16).

[35] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In: *International conference on machine learning*. PMLR. 2017, pp. 933–941 (cit. on p. 9).

[36] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. In: *arXiv preprint arXiv:2208.07339* (2022) (cit. on p. 16).

[37] Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 8-bit Optimizers via Block-wise Quantization. In: *9th International Conference on Learning Representations, ICLR* (2022) (cit. on p. 16).

[38] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In: *arXiv preprint arXiv:2305.14314* (2023) (cit. on p. 16).

[39] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186 (cit. on p. 36).

[40] Thomas G. Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. In: *Journal of artificial intelligence research* 2 (1994), pp. 263–286 (cit. on p. 37).

[41] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 2793–2803 (cit. on p. 13).

[42] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In: *International Conference on Learning Representations*. 2020 (cit. on p. 10).

[43] T. Dunning and New Mexico State University. Computing Research Laboratory. Statistical Identification of Language. Memoranda in computer and cognitive science. Computing Research Laboratory, New Mexico State University, 1994 (cit. on p. 7).

[44] N Elhage, N Nanda, C Olsson, et al. A mathematical framework for transformer circuits. In: *Transformer Circuits Thread* (2021) (cit. on p. 13).

[45] Jeffrey L Elman. Finding structure in time. In: *Cognitive science* 14.2 (1990), pp. 179–211 (cit. on p. 8).

[46] Tyna Eloundou, Sam Manning, Pamela Mishkin, and Daniel Rock. Gpts are gpts: An early look at the labor market impact potential of large language models. In: *arXiv preprint arXiv:2303.10130* (2023) (cit. on p. 1).

[47] Guy Emerson, Natalie Schluter, Gabriel Stanovsky, et al., eds. Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022). Seattle, United States: Association for Computational Linguistics, July 2022 (cit. on p. 25).

[48] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical Neural Story Generation. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 889–898 (cit. on p. 17).

[49] Asier Gutiérrez Fandiño, Jordi Armengol Estapé, Marc Pàmies, et al. MarIA: Spanish Language Models. In: *Procesamiento del Lenguaje Natural* (2022) (cit. on p. 46).

[50] Asier Gutiérrez Fandiño, Jordi Armengol Estapé, Marc Pàmies, et al. MarIA: Spanish Language Models. In: *Procesamiento del Lenguaje Natural* 68 (2022) (cit. on p. 37).

[51] Rudolph Flesch. A new readability yardstick. In: *Journal of applied psychology* (1948) (cit. on p. 49).

[52] Leon Fröhling and Arkaitz Zubiaga. Feature-based detection of automated language models: tackling GPT-2, GPT-3 and Grover. en. In: *PeerJ Computer Science* 7 (Apr. 2021) (cit. on p. 22).

[53] Daniel Y. Fu, Tri Dao, Khaled K. Saab, et al. Hungry Hungry Hippos: Towards Language Modeling with State Space Models. In: *International Conference on Learning Representations*. 2023 (cit. on p. 9).

[54] Matthias Gallé, Jos Rozen, Germán Kruszewski, and Hady ElSahar. Unsupervised and Distributional Detection of Machine-Generated Text. In: *ArXiv* abs/2111.02878 (2021) (cit. on p. 22).

[55] Deep Ganguli, Liane Lovitt, Jackson Kernion, et al. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. In: *arXiv preprint arXiv:2209.07858* (2022) (cit. on p. 20).

[56] Leo Gao, Stella Biderman, Sid Black, et al. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. In: *CoRR* abs/2101.00027 (2021). arXiv: `2101.00027` (cit. on p. 16).

[57] Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. GLTR: Statistical Detection and Visualization of Generated Text. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. 2019, pp. 111–116 (cit. on p. 22).

[58] Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer Feed-Forward Layers Are Key-Value Memories. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2021, pp. 5484–5495 (cit. on p. 13).

[59] Bilal Ghanem, Simone Paolo Ponzetto, Paolo Rosso, and Francisco Rangel. FakeFlow: Fake News Detection by Modeling the Flow of Affective Information. In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. 2021, pp. 679–689 (cit. on p. 1).

[60] Neil M. Goldman. Computer generation of natural language from a deep conceptual base. In: 1974 (cit. on p. 6).

[61] M. Dolores González, Eugenio Martínez-Cámara, Maria Martín-Valdivia, and L. López. Cross-Domain Sentiment Analysis Using Spanish Opinionated Words. In: vol. 8455. June 2014, pp. 214–219 (cit. on p. 28).

[62] Craig S. Greenberg, Vincent M. Stanford, Alvin F. Martin, et al. The 2012 NIST speaker recognition evaluation. In: *Interspeech*. 2017 (cit. on p. 25).

[63] Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, et al. More than you've asked for: A Comprehensive Analysis of Novel Prompt Injection Threats to Application-Integrated Large Language Models. In: *arXiv preprint arXiv:2302.12173* (2023) (cit. on p. 20).

[64] Chenxi Gu, Chengsong Huang, Xiaoqing Zheng, Kai-Wei Chang, and Cho-Jui Hsieh. Watermarking Pre-trained Language Models with Backdooring. In: *arXiv preprint arXiv:2210.07543* (2022) (cit. on p. 23).

[65] Jesus Cordova Guerrero and Izzat Alsmadi. Synthetic Text Detection: Systemic Literature Review. In: *ArXiv* abs/2210.06336 (2022) (cit. on p. 23).

[66] Sylvain Gugger, Lysandre Debut, Thomas Wolf, et al. Accelerate: Training and inference at scale made simple, efficient and adaptable. `https://github.com/huggingface/accelerate`. 2022 (cit. on p. 15).

[67] Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, et al. XL-Sum: Large-Scale Multilingual Abstractive Summarization for 44 Languages. In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Online: Association for Computational Linguistics, Aug. 2021, pp. 4693–4703 (cit. on p. 28).

[68] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778 (cit. on p. 11).

[69] Pengcheng He, Jianfeng Gao, and Weizhu Chen. DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing. In: *The Eleventh International Conference on Learning Representations*. 2023 (cit. on pp. 36, 37, 46).

[70] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. In: *Neural computation* 9.8 (1997), pp. 1735–1780 (cit. on pp. 9, 36).

[71] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, et al. Training compute-optimal large language models. In: *arXiv preprint arXiv:2203.15556* (2022) (cit. on p. 13).

[72] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The Curious Case of Neural Text Degeneration. In: *International Conference on Learning Representations*. 2020 (cit. on pp. 17, 29).

[73] Yue Hu and Xiaojun Wan. Automatic Generation of Related Work Sections in Scientific Papers: An Optimization Approach. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1624–1633 (cit. on p. 1).

[74] Xiao Shi Huang, Felipe Perez, Jimmy Ba, and Maksims Volkovs. Improving Transformer Optimization Through Better Initialization. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, July 2020, pp. 4475–4483 (cit. on p. 10).

[75] Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. Automatic Detection of Generated Text is Easiest when Humans are Fooled. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020, pp. 1808–1822 (cit. on p. 22).

[76] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of Tricks for Efficient Text Classification. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, Apr. 2017, pp. 427–431 (cit. on p. 27).

[77] B. H. Juang and L. R. Rabiner. Hidden Markov Models for Speech Recognition. In: *Technometrics* 33.3 (1991), pp. 251–272 (cit. on p. 7).

[78] John Jumper, Richard Evans, Alexander Pritzel, et al. Highly accurate protein structure prediction with AlphaFold. In: *Nature* 596.7873 (2021), pp. 583–589 (cit. on p. 10).

[79] Jared Kaplan, Sam McCandlish, Tom Henighan, et al. Scaling laws for neural language models. In: *arXiv preprint arXiv:2001.08361* (2020) (cit. on p. 13).

[80] Yury Kashnitsky, Drahomira Herrmannova, Anita de Waard, et al. Overview of the DAGPap22 Shared Task on Detecting Automatically Generated Scientific Papers. In: *Proceedings of the Third Workshop on Scholarly Document Processing*. Gyeongju, Republic of Korea: Association for Computational Linguistics, Oct. 2022, pp. 210–213 (cit. on p. 2).

[81] Yury Kashnitsky, Drahomira Herrmannova, Anita de Waard, et al. Overview of the DAGPap22 Shared Task on Detecting Automatically Generated Scientific Papers. In: *Proceedings of the Third Workshop on Scholarly Document Processing*. 2022, pp. 210–213 (cit. on pp. 2, 25).

[82] Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, et al. ChatGPT for good? On opportunities and challenges of large language models for education. In: *Learning and Individual Differences* (2023), p. 102274 (cit. on p. 2).

[83] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through Memorization: Nearest Neighbor Language Models. In: *International Conference on Learning Representations*. 2019 (cit. on p. 15).

[84] John Kirchenbauer, Jonas Geiping, Yuxin Wen, et al. A watermark for large language models. In: *arXiv preprint arXiv:2301.10226* (2023) (cit. on pp. 2, 23).

[85] Vijay Anand Korthikanti, Jared Casper, Sangkug Lym, et al. Reducing activation recomputation in large transformer models. In: *Proceedings of Machine Learning and Systems* 5 (2023) (cit. on p. 15).

[86] Taku Kudo. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 66–75 (cit. on p. 17).

[87] Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 66–71 (cit. on p. 17).

[88] Faisal Ladhak, Esin Durmus, Claire Cardie, and Kathleen McKeown. WikiLingua: A New Benchmark Dataset for Cross-Lingual Abstractive Summarization. In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, Nov. 2020, pp. 4034–4048 (cit. on p. 28).

[89] K. Lari and S.J. Young. The estimation of stochastic context-free grammars using the Inside-Outside algorithm. In: *Computer Speech & Language* 4.1 (1990), pp. 35–56 (cit. on p. 7).

[90] Hugo Laurençon, Lucile Saulnier, Thomas Wang, et al. The BigScience ROOTS Corpus: A 1.6TB Composite Multilingual Dataset. In: *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2022 (cit. on p. 16).

[91] Yann LeCun, Bernhard Boser, John Denker, et al. Handwritten digit recognition with a back-propagation network. In: *Advances in neural information processing systems* 2 (1989) (cit. on p. 36).

[92] Tao Lei, Yu Zhang, Sida I Wang, Hui Dai, and Yoav Artzi. Simple Recurrent Units for Highly Parallelizable Recurrence. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2018, pp. 4470–4481 (cit. on p. 9).

[93] Angela Leis, Francesco Ronzano, Miguel A Mayer, Laura I Furlong, and Ferran Sanz. Detecting Signs of Depression in Tweets in Spanish: Behavioral and Linguistic Analysis. In: *J Med Internet Res* 21.6 (June 2019), e14199 (cit. on p. 28).

[94] Pericles Lewis. The Cambridge introduction to modernism. Cambridge University Press, 2007 (cit. on p. 6).

[95] Conglong Li, Zhewei Yao, Xiaoxia Wu, Minjia Zhang, and Yuxiong He. DeepSpeed Data Efficiency: Improving Deep Learning Model Quality and Training Efficiency via Efficient Data Sampling and Routing. In: *arXiv preprint arXiv:2212.03597* (2022) (cit. on p. 15).

[96] Raymond Li, Loubna Ben Allal, Yangtian Zi, et al. StarCoder: may the source be with you! In: *ArXiv* abs/2305.06161 (2023) (cit. on p. 16).

[97] Percy Liang, Rishi Bommasani, Tony Lee, et al. Holistic evaluation of language models. In: *arXiv preprint arXiv:2211.09110* (2022) (cit. on p. 10).

[98] Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, et al. Few-shot Learning with Multilingual Generative Language Models. In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 9019–9052 (cit. on p. 30).

[99] Yiheng Liu, Tianle Han, Siyuan Ma, et al. Summary of chatgpt/gpt-4 research and perspective towards the future of large language models. In: *arXiv preprint arXiv:2304.01852* (2023) (cit. on p. 1).

[100] Yinhan Liu, Myle Ott, Naman Goyal, et al. Roberta: A robustly optimized bert pretraining approach. In: *arXiv preprint arXiv:1907.11692* (2019) (cit. on p. 36).

[101] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. In: *Journal of machine learning research* 9.11 (2008) (cit. on p. 30).

[102] A.A. Markov. An example of statistical investigation of the text Eugene Onegin concerning the connection of samples in chains. In: *(In Russian.) Bulletin of the Imperial Academy of Sciences of St. Petersburg* 7.3 (1913), pp. 153–162 (cit. on p. 6).

[103] Antonis Maronikolakis, Hinrich Schütze, and Mark Stevenson. Identifying Automatically Generated Headlines using Transformers. In: *Proceedings of the Fourth Workshop on NLP for Internet Freedom: Censorship, Disinformation, and Propaganda*. 2021, pp. 1–6 (cit. on p. 22).

[104] Julian McAuley and Jure Leskovec. Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text. In: *Proceedings of the 7th ACM Conference on Recommender Systems*. RecSys '13. Hong Kong, China: Association for Computing Machinery, 2013, 165–172 (cit. on p. 28).

[105] Paulius Micikevicius, Sharan Narang, Jonah Alben, et al. Mixed Precision Training. In: *International Conference on Learning Representations*. 2017 (cit. on p. 16).

[106] Paulius Micikevicius, Dusan Stosic, Neil Burgess, et al. FP8 formats for deep learning. In: *arXiv preprint arXiv:2209.05433* (2022) (cit. on p. 16).

[107] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In: *arXiv preprint arXiv:2301.11305* (2023) (cit. on pp. 2, 22).

[108] Manuel Montes-y-Gómez, Julio Gonzalo, Francisco Rangel, et al., eds. Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2022) co-located with the Conference of the Spanish Society for Natural Language Processing (SEPLN 2022), A Coruña, Spain, September 20, 2022. Vol. 3202. CEUR Workshop Proceedings. CEUR-WS.org, 2022 (cit. on p. 25).

[109] Michael C Mozer. A focused backpropagation algorithm for temporal pattern recognition. In: *Backpropagation: Theory, architectures, and applications* 137 (1995) (cit. on p. 9).

[110] Thomas Mueller, Guillermo Pérez-Torró, and Marc Franco-Salvador. Few-Shot Learning with Siamese Networks and Label Tuning. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2022, pp. 8532–8545 (cit. on p. 38).

[111] Niklas Muennighoff, Thomas Wang, Lintang Sutawika, et al. Crosslingual Generalization through Multitask Finetuning. 2022 (cit. on p. 16).

[112] Kenton Murray and David Chiang. Correcting Length Bias in Neural Machine Translation. In: *Proceedings of the Third Conference on Machine Translation: Research Papers*. Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 212–223 (cit. on p. 17).

[113] Ibrahim Naji. TSATC: Twitter Sentiment Analysis Training Corpus. In: *thinknook*. 2012 (cit. on p. 28).

[114] Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In: *ArXiv* abs/1808.08745 (2018) (cit. on p. 28).

[115] Hermann Ney, Ute Essen, and Reinhard Kneser. On structuring probabilistic dependences in stochastic language modelling. In: *Computer Speech & Language* 8.1 (1994), pp. 1–38 (cit. on p. 7).

[116] nostalgebraist. interpreting GPT: the logit lens. In: *LessWrong* (2020) (cit. on p. 13).

[117] OpenAI. 2023 (cit. on p. 2).

[118] Long Ouyang, Jeffrey Wu, Xu Jiang, et al. Training language models to follow instructions with human feedback. In: *Advances in Neural Information Processing Systems*. 2022 (cit. on pp. 1, 28).

[119] Jing Pan, Tao Lei, Kwangyoun Kim, Kyu J Han, and Shinji Watanabe. Sru++: Pioneering fast recurrence with attention for speech recognition. In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2022, pp. 7872–7876 (cit. on p. 9).

[120] P Panicheva, J Cardiff, and P Rosso. Personal sense and idiolect: Combining authorship attribution and opinion analysis. In: *Proceedings of the 7th International Conference on Language Resources and Evaluation, LREC 2010*. 2010, pp. 1134–1137 (cit. on p. 23).

[121] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a Method for Automatic Evaluation of Machine Translation. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, July 2002, pp. 311–318 (cit. on p. 19).

[122] Adam Paszke, Sam Gross, Francisco Massa, et al. Pytorch: An imperative style, high-performance deep learning library. In: *Advances in neural information processing systems* 32 (2019) (cit. on p. 16).

[123] F. Pedregosa, G. Varoquaux, A. Gramfort, et al. Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. on p. 38).

[124] Bo Peng, Eric Alcaide, Quentin Anthony, et al. RWKV: Reinventing RNNs for the Transformer Era. In: *arXiv preprint arXiv:2305.13048* (2023) (cit. on p. 9).

[125] Ethan Perez, Saffron Huang, Francis Song, et al. Red Teaming Language Models with Language Models. In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 2022, pp. 3419–3448 (cit. on p. 20).

[126] Fábio Perez and Ian Ribeiro. Ignore Previous Prompt: Attack Techniques For Language Models. In: *NeurIPS ML Safety Workshop*. 2022 (cit. on p. 20).

[127] Krishna Pillutla, Lang Liu, John Thickstun, et al. MAUVE scores for generative models: Theory and practice. In: *arXiv preprint arXiv:2212.14578* (2022) (cit. on p. 32).

[128] Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, et al. MAUVE: Measuring the Gap Between Neural Text and Human Text using Divergence Frontiers. In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan. 2021 (cit. on pp. 19, 32).

[129] Juan Pizarro. Using N-grams to detect Bots on Twitter. In: *Conference and Labs of the Evaluation Forum*. 2019 (cit. on p. 38).

[130] Michael Poli, Stefano Massaroli, Eric Nguyen, et al. Hyena Hierarchy: Towards Larger Convolutional Language Models. In: *arXiv preprint arXiv:2302.10866* (2023) (cit. on p. 9).

[131] Stanislas Polu and Ilya Sutskever. Generative language modeling for automated theorem proving. In: *arXiv preprint arXiv:2009.03393* (2020) (cit. on p. 10).

[132] Ofir Press, Noah Smith, and Mike Lewis. Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation. In: *International Conference on Learning Representations*. 2021 (cit. on p. 15).

[133] Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. BPE-Dropout: Simple and Effective Subword Regularization. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, pp. 1882–1892 (cit. on p. 17).

[134] Alec Radford, Jeffrey Wu, Rewon Child, et al. Language models are unsupervised multitask learners. In: *OpenAI blog* 1.8 (2019), p. 9 (cit. on pp. 14, 22, 30, 36).

[135] Colin Raffel, Noam Shazeer, Adam Roberts, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. In: *The Journal of Machine Learning Research* 21.1 (2020), pp. 5485–5551 (cit. on pp. 10, 16).

[136] Francisco Rangel, Marc Franco-Salvador, and Paolo Rosso. A low dimensionality representation for language variety identification. In: *Computational Linguistics and Intelligent Text Processing: 17th International Conference, CICLing 2016, Konya, Turkey, April 3–9, 2016, Revised Selected Papers, Part II 17*. Springer. 2018, pp. 156–169 (cit. on p. 38).

[137] Jie Ren, Samyam Rajbhandari, Reza Yazdani Aminabadi, et al. ZeRO-Offload: Democratizing Billion-Scale Model Training. In: *CoRR* abs/2101.06840 (2021). arXiv: `2101.06840` (cit. on p. 15).

[138] Juan Rodriguez, Todd Hay, David Gros, Zain Shamsi, and Ravi 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*. 2022, pp. 1213–1233 (cit. on pp. 2, 22).

[139] R. Rosenfeld. Two decades of statistical language modeling: where do we go from here? In: *Proceedings of the IEEE* 88.8 (2000), pp. 1270–1278 (cit. on p. 6).

[140] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. In: *nature* 323.6088 (1986), pp. 533–536 (cit. on p. 8).

[141] Vinu Sankar Sadasivan, Aounon Kumar, Sriram Balasubramanian, Wenxiao Wang, and Soheil Feizi. Can AI-Generated Text be Reliably Detected? In: *arXiv preprint arXiv:2303.11156* (2023) (cit. on p. 23).

[142] Victor Sanh, Albert Webson, Colin Raffel, et al. Multitask Prompted Training Enables Zero-Shot Task Generalization. 2021. arXiv: `2110.08207 [cs.LG]` (cit. on p. 16).

[143] Upendra Sapkota, Thamar Solorio, Manuel Montes, Steven Bethard, and Paolo Rosso. Cross-topic authorship attribution: Will out-of-topic data help? In: *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. 2014, pp. 1228–1237 (cit. on p. 23).

[144] Elvis Saravia. Prompt Engineering Guide. In: *https://github.com/dair-ai/Prompt-Engineering-Guide* (Dec. 2022) (cit. on p. 20).

[145] Areg Mikael Sarvazyan, José Ángel González, Marc Franco Salvador, et al. Overview of AuTexTification at IberLEF 2023: Detection and Attribution of Machine-Generated Text in Multiple Domains. In: *Procesamiento del Lenguaje Natural*. Jaén, Spain, Sept. 2023 (cit. on p. 43).

[146] Areg Mikael Sarvazyan, José Ángel González, Marc Franco-Salvador, and Paolo Rosso. Supervised Machine-Generated Text Detectors: Family and Scale Matters. In: *Information Access Evaluation meets Multilinguality, Multimodality, and Visualization*. Springer International Publishing, 2023 (cit. on pp. 50, 53).

[147] Teven Le Scao, Angela Fan, Christopher Akiki, et al. Bloom: A 176b-parameter open-access multilingual language model. In: *arXiv preprint arXiv:2211.05100* (2022) (cit. on pp. 1, 14, 15, 28, 30, 46).

[148] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. In: *IEEE transactions on Signal Processing* 45.11 (1997), pp. 2673–2681 (cit. on pp. 9, 37).

[149] Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. MLSUM: The Multilingual Summarization Corpus. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 8051–8067 (cit. on p. 28).

[150] Elizabeth Seger, Aviv Ovadya, Ben Garfinkel, Divya Siddarth, and Allan Dafoe. Democratising AI: Multiple Meanings, Goals, and Methods. In: *arXiv preprint arXiv:2303.12642* (2023) (cit. on p. 1).

[151] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 1715–1725 (cit. on p. 17).

[152] Tatiana Shamardina, Vladislav Mikhailov, Daniil Chernianskii, et al. Findings of the the ruatd shared task 2022 on artificial text detection in russian. In: *arXiv preprint arXiv:2206.01583* (2022) (cit. on pp. 2, 25).

[153] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-Attention with Relative Position Representations. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics. 2018 (cit. on p. 14).

[154] Prasha Shrestha, Sebastian Sierra, Fabio A González, et al. Convolutional neural networks for authorship attribution of short texts. In: *Proceedings of the 15th conference of the European chapter of the association for computational linguistics: Volume 2, short papers*. 2017, pp. 669–674 (cit. on p. 23).

[155] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, et al. Model Dementia: Generated Data Makes Models Forget. In: *arXiv preprint arXiv:2305.17493* (2023) (cit. on p. 2).

[156] Irene Solaiman, Miles Brundage, Jack Clark, et al. Release strategies and the social impacts of language models. In: *arXiv preprint arXiv:1908.09203* (2019) (cit. on p. 22).

[157] Shane Storks, Qiaozi Gao, and Joyce Y Chai. Recent advances in natural language inference: A survey of benchmarks, resources, and approaches. In: *arXiv preprint arXiv:1904.01172* (2019) (cit. on p. 19).

[158] Jianlin Su, Yu Lu, Shengfeng Pan, et al. Roformer: Enhanced transformer with rotary position embedding. In: *arXiv preprint arXiv:2104.09864* (2021) (cit. on p. 14).

[159] Yixuan Su and Nigel Collier. Contrastive Search Is What You Need For Neural Text Generation. In: *Transactions on Machine Learning Research* (2023) (cit. on p. 17).

[160] Yixuan Su, Tian Lan, Yan Wang, et al. A Contrastive Framework for Neural Text Generation. In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho. 2022 (cit. on pp. 17, 18).

[161] Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, et al. Dataset Cartography: Mapping and Diagnosing Datasets with Training Dynamics. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2020, pp. 9275–9293 (cit. on p. 33).

[162] Yi Tay, Mostafa Dehghani, Samira Abnar, et al. Long Range Arena : A Benchmark for Efficient Transformers. In: *International Conference on Learning Representations*. 2021 (cit. on p. 9).

[163] MosaicML NLP Team. Introducing MPT-7B: A New Standard for Open-Source, Commercially Usable LLMs. Accessed: 2023-05-05. 2023. URL: `www.mosaicml.com/blog/mpt-7b` (visited on May 5, 2023) (cit. on p. 15).

[164] Julien Tourille, Babacar Sow, and Adrian Popescu. Automatic Detection of Bot-Generated Tweets. In: *Proceedings of the 1st International Workshop on Multimedia AI against Disinformation*. 2022, 44–51 (cit. on p. 2).

[165] Hugo Touvron, Thibaut Lavril, Gautier Izacard, et al. Llama: Open and efficient foundation language models. In: *arXiv preprint arXiv:2302.13971* (2023) (cit. on pp. 1, 14, 28).

[166] Adaku Uchendu, Thai Le, and Dongwon Lee. Attribution and Obfuscation of Neural Text Authorship: A Data Mining Perspective. In: *SIGKDD Explor. Newsl.* 25.1 (July 2023), 1–18 (cit. on p. 23).

[167] Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. Authorship Attribution for Neural Text Generation. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 8384–8395 (cit. on pp. 2, 24).

[168] Adaku Uchendu, Zeyu Ma, Thai Le, Rui Zhang, and Dongwon 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*. Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 2001–2016 (cit. on p. 24).

[169] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. Attention is all you need. In: *Advances in neural information processing systems* 30 (2017) (cit. on pp. 10, 11, 22, 36).

[170] Veniamin Veselovsky, Manoel Horta Ribeiro, and Robert West. Artificial Artificial Artificial Intelligence: Crowd Workers Widely Use Large Language Models for Text Production Tasks. In: *arXiv preprint arXiv:2306.07899* (2023) (cit. on p. 2).

[171] Ben Wang and Aran Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`. May 2021 (cit. on p. 14).

[172] Guibin Wang, YiSong Lin, and Wei Yi. Kernel Fusion: An Effective Method for Better Power Efficiency on Multithreaded GPU. In: *2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*. 2010, pp. 344–350 (cit. on p. 16).

[173] Jindong Wang, Xixu Hu, Wenxin Hou, et al. On the robustness of chatgpt: An adversarial and out-of-distribution perspective. In: *arXiv preprint arXiv:2302.12095* (2023) (cit. on p. 20).

[174] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How Does LLM Safety Training Fail? 2023 (cit. on p. 20).

[175] Sean Welleck, Ilia Kulikov, Stephen Roller, et al. Neural Text Generation With Unlikelihood Training. In: *International Conference on Learning Representations*. 2020 (cit. on p. 18).

[176] Jules White, Quchen Fu, Sam Hays, et al. A prompt pattern catalog to enhance prompt engineering with chatgpt. In: *arXiv preprint arXiv:2302.11382* (2023) (cit. on p. 20).

[177] David Gray Widder, Dawn Nafus, Laura Dabbish, and James Herbsleb. Limits and Possibilities for "Ethical AI" in Open Source: A Study of Deepfakes. In: *2022 ACM Conference on Fairness, Accountability, and Transparency*. 2022, pp. 2035–2046 (cit. on p. 2).

[178] Jason Williams, Antoine Raux, Deepak Ramachandran, and Alan Black. The Dialog State Tracking Challenge. In: *Proceedings of the SIGDIAL 2013 Conference*. Metz, France: Association for Computational Linguistics, Aug. 2013, pp. 404–413 (cit. on p. 25).

[179] Ronald J Williams. Gradient-based learning algorithms for recurrent networks. In: *Back-propagation: Theory, Architectures and Applications* (1989) (cit. on p. 9).

[180] Thomas Wolf, Lysandre Debut, Victor Sanh, et al. Transformers: State-of-the-art natural language processing. In: *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*. 2020, pp. 38–45 (cit. on pp. 1, 38).

[181] Thomas Wolf, Lysandre Debut, Victor Sanh, et al. Transformers: State-of-the-Art Natural Language Processing. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 2020, pp. 38–45 (cit. on p. 46).

[182] Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. Pay Less Attention with Lightweight and Dynamic Convolutions. In: *International Conference on Learning Representations*. 2019 (cit. on p. 9).

[183] Shijie Wu, Ozan Irsoy, Steven Lu, et al. Bloomberggpt: A large language model for finance. In: *arXiv preprint arXiv:2303.17564* (2023) (cit. on pp. 1, 13).

[184] Xiaoxia Wu, Cheng Li, Reza Yazdani Aminabadi, Zhewei Yao, and Yuxiong He. Understanding INT4 Quantization for Transformer Models: Latency Speedup, Composability, and Failure Cases. In: *arXiv preprint arXiv:2301.12017* (2023) (cit. on p. 16).

[185] Ruibin Xiong, Yunchang Yang, Di He, et al. On layer normalization in the transformer architecture. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 10524–10533 (cit. on p. 14).

[186] Rowan Zellers, Ari Holtzman, Hannah Rashkin, et al. Defending against neural fake news. In: *Advances in neural information processing systems* (2019) (cit. on pp. 2, 22, 36).

[187] Susan Zhang, Stephen Roller, Naman Goyal, et al. Opt: Open pre-trained transformer language models. In: *arXiv preprint arXiv:2205.01068* (2022) (cit. on pp. 28, 36).

[188] Wei Emma Zhang, Quan Z Sheng, Ahoud Alhazmi, and Chenliang Li. Adversarial attacks on deep-learning models in natural language processing: A survey. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 11.3 (2020), pp. 1–41 (cit. on p. 20).

[189] Xinyang Zhang, Yury Malkov, Omar Florez, et al. TwHIN-BERT: A Socially-Enriched Pre-trained Language Model for Multilingual Tweet Representations. In: *arXiv preprint arXiv:2209.07562* (2022) (cit. on p. 36).

[190] Yaoming Zhu, Sidi Lu, Lei Zheng, et al. Texygen: A Benchmarking Platform for Text Generation Models. In: *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. SIGIR '18. Ann Arbor, MI, USA: Association for Computing Machinery, 2018, 1097–1100 (cit. on p. 18).

A Additional Dataset Plots

In the following sections we provide additional plots to further understand the distributions of the texts in the AuTexTification 2023 dataset.

A.1 First Generation Efforts

Figure A.1 presents the resulting plots of z-score normalized evaluation metrics (i.e. repetition, diversity, stop-word and symbol ratios, self-BLEU, perplexity) projected to two dimensions with t-SNE for the first dataset using small models (see Section 4.3.3).

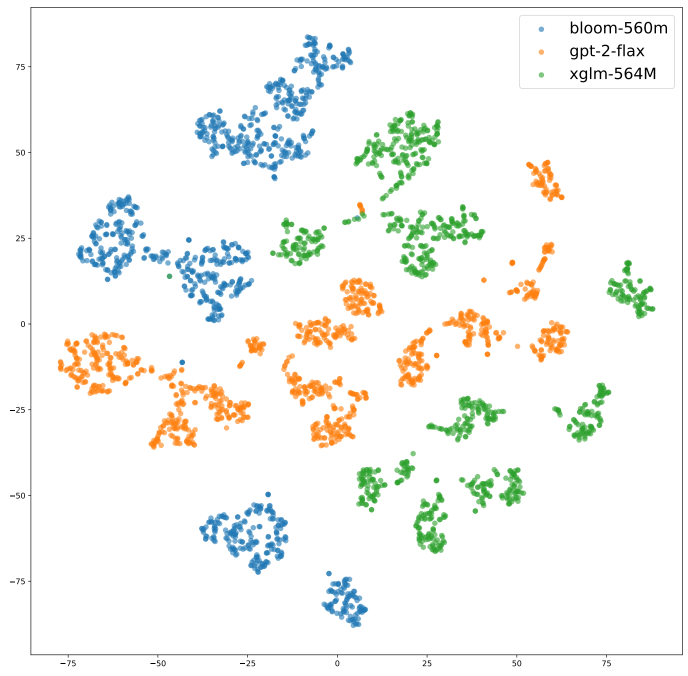
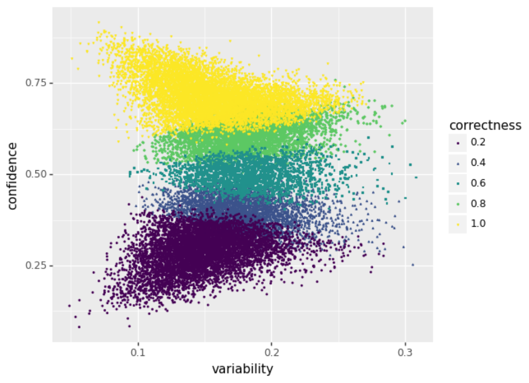


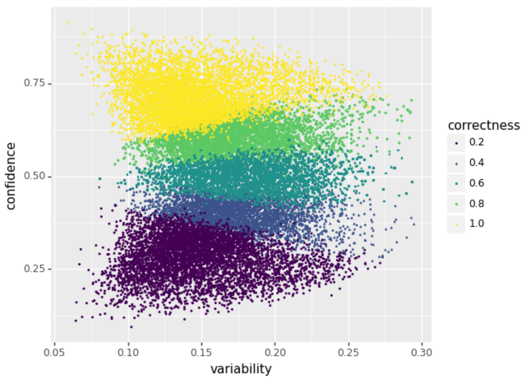
Fig. A.1.: t-SNE plots for MGT in reviews using small models.

A.2 Additional Data Maps

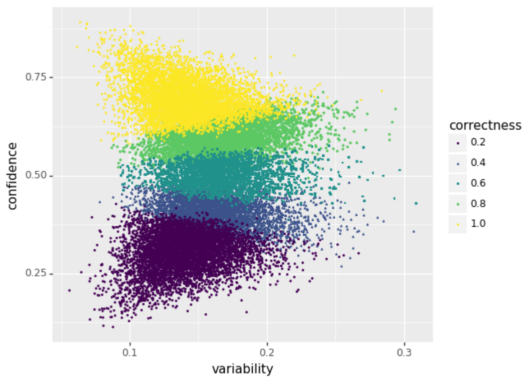
Here we present additional zero-shot data maps for various partitions of the AuTex-Tification 2023 dataset. Figure A.2 shows data maps partitioned by training or test split, as well as language. Additionally, figures A.3 and A.4 present maps for Subtask 1 partitioned by text generation model in English and Spanish, respectively. Finally, figure A.5 illustrates data maps in Spanish grouped by domain.



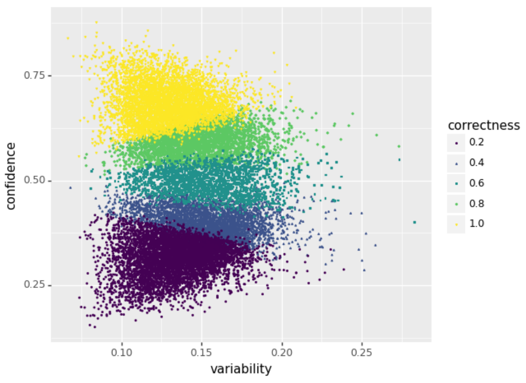
(a) Train split in English.



(b) Test split in English.

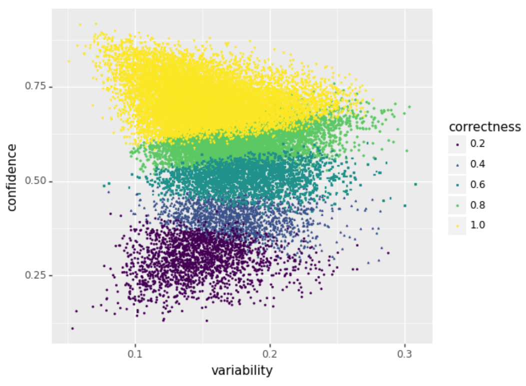


(c) Train split in Spanish.

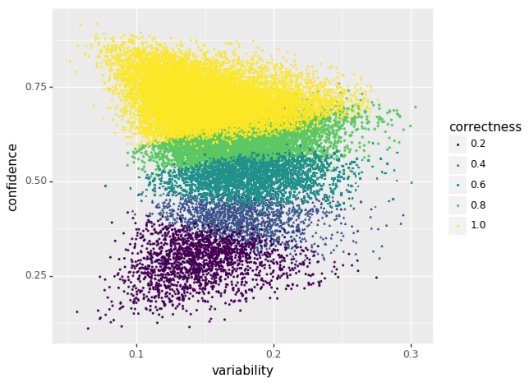


(d) Test split in Spanish.

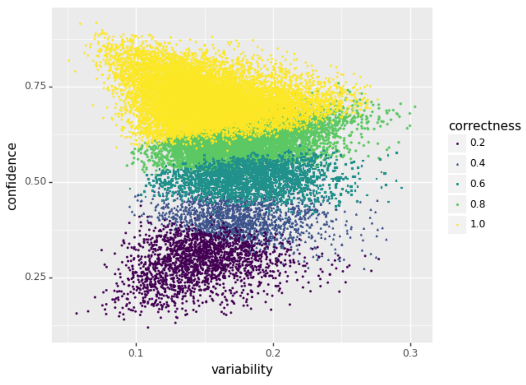
Fig. A.2.: Data maps for Subtask 1 per split and language.



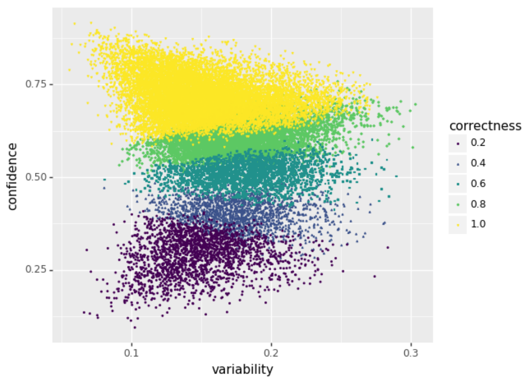
(a) MGT generated by BLOOM-1B1.



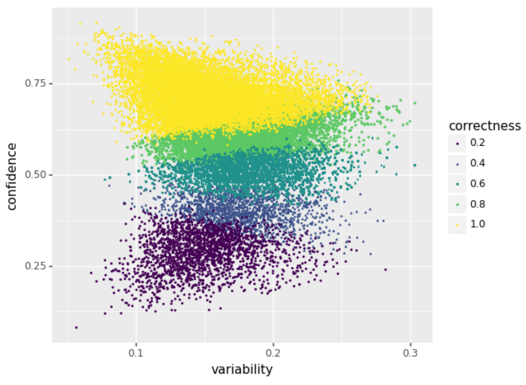
(b) MGT generated by BLOOM-3B.



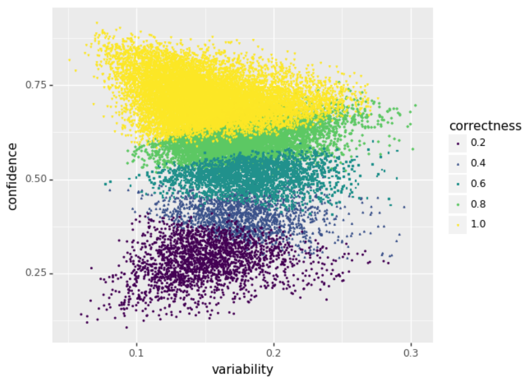
(c) MGT generated by BLOOM-7B1.



(d) MGT generated by babbage.

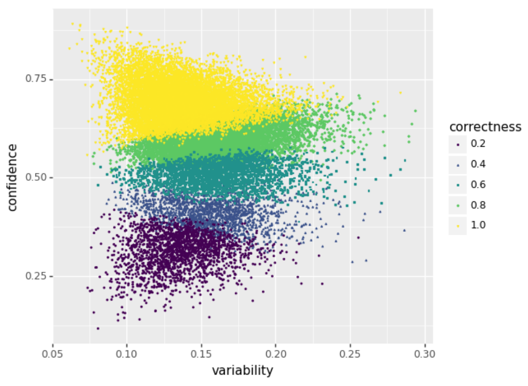


(e) MGT generated by curie.

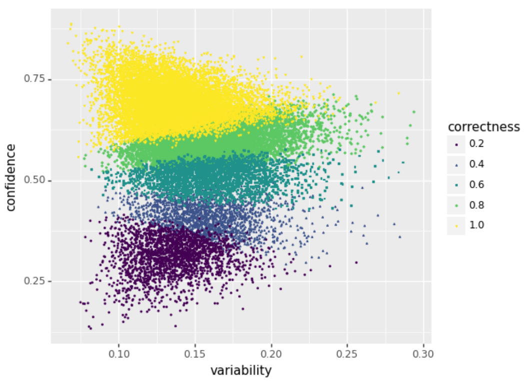


(f) MGT generated by davinci.

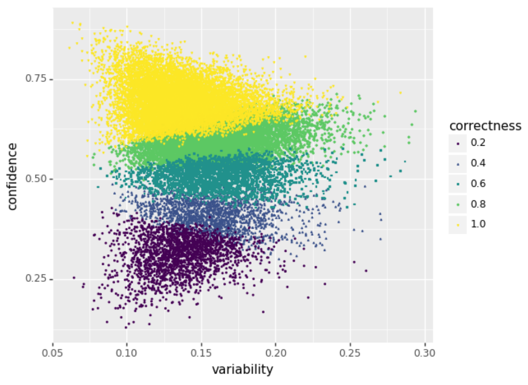
Fig. A.3.: Data maps for Subtask 1 in English with different generation models.



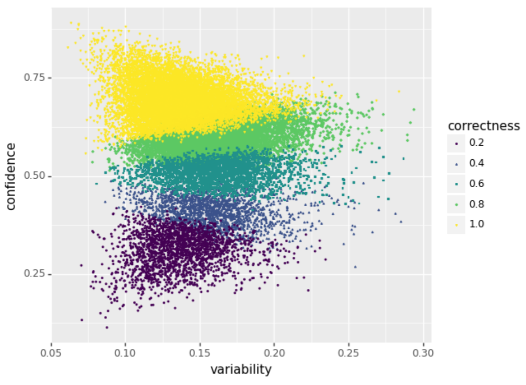
(a) MGT generated by BLOOM-1B1.



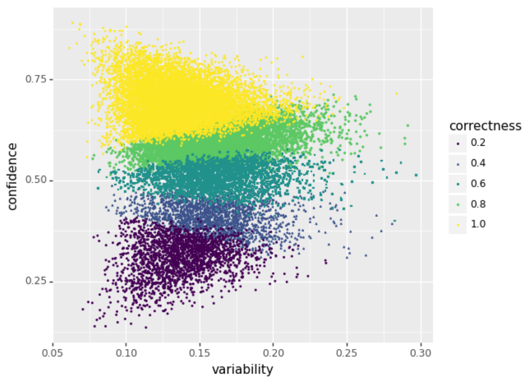
(b) MGT generated by BLOOM-3B.



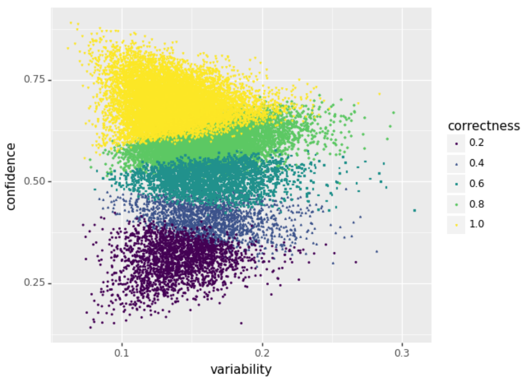
(c) MGT generated by BLOOM-7B1.



(d) MGT generated by babbage.

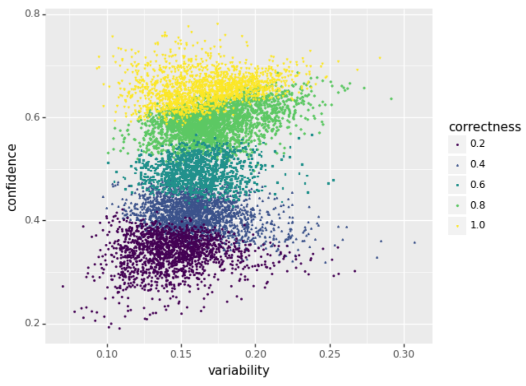


(e) MGT generated by curie.

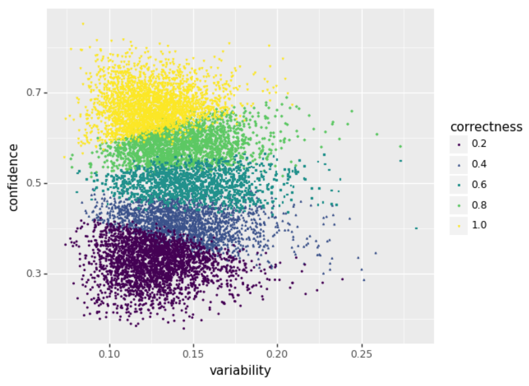


(f) MGT generated by davinci.

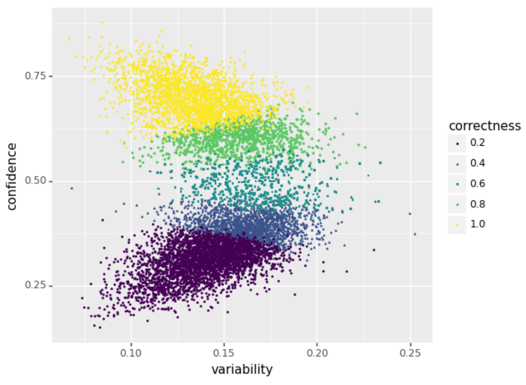
Fig. A.4.: Data maps for Subtask 1 in Spanish with different generation models.



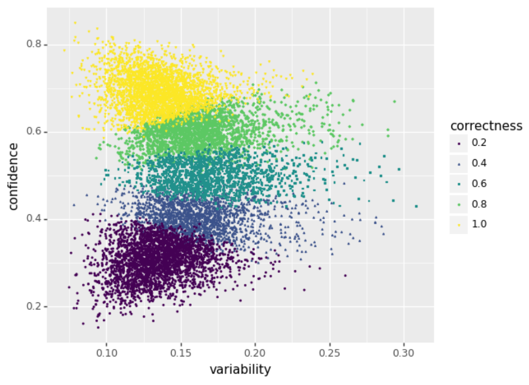
(a) Legal.



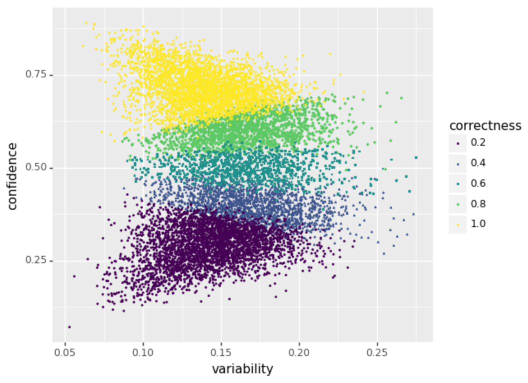
(b) News.



(c) Reviews.



(d) Wiki-How.



(e) Tweets.

Fig. A.5.: Data maps for Subtask 1 in Spanish grouped by domain.