ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230641

CICERO: A GPT2-Based Writing Assistant to Investigate the Effectiveness of Specialized LLMs' Applications in E-Justice

Marco Calamo, Francesca De Luzi[*], Mattia Macrì, Tommaso Mencattini and Massimo Mecella

Sapienza Università di Roma, Rome, Italy

Abstract. Does it make sense to develop specialized writing assistants in the era of LLMs - Large Language Models? In this paper, we present CICERO, a specialized writing assistant we have developed for writing (pieces of) sentences in the Italian legal system. Our proposed solution involves fine-tuning a transformer on a pre-processed corpus of Italian civil judgments, resulting in a *novel* language model that can be deployed as a writing assistant for legal users to improve text writing efficiency. The model can also be further fine-tuned for use in other law-related natural language processing tasks. The experimental validation of CICERO allows us also to draw interesting insights on the meaningful, if any, of developing specialized tools for assisting specific classes of users and knowledge workers, in an era in which we witness the widespread adoption of LLMs.

1 Introduction

Enhancing the efficiency and transparency of government operations, by promoting citizen participation and civil servant effectiveness in public management, are ultimately the primary goals of implementing *e-government* [20]. However, such an implementation poses several challenges, including the requirement for robust technological infrastructure and sufficient staff training, as well as ensuring the security and confidentiality of citizens' data [26]. The concept of e-government emerged in the 1990s and early 2000s, utilizing information and communication technologies (ICT) to provide and improve public services to citizens and businesses. Nowadays, the use of ICT technologies can be further enhanced by integrating Artificial Intelligence (AI) tools into public administrations. This integration offers the potential for improving the efficiency and effectiveness of services. The models and methods of AI can be used to automate administrative processes, including the verification of application compliance, automatic document generation, and workflow management. This can lead to reduced processing time and increased efficiency. However, the implementation of AI in e-government requires a holistic and multidisciplinary approach, involving not only technology but also governance, ethics, public participation, and openness.

In this context, the authors are involved in the *"Giustizia AGILE"* project [1] aimed at improving the digital transition in justice management, enhancing the balance between quality and efficiency of judicial decisions, providing greater organizational support for computerization and telematization of judicial offices, and facilitating concrete change management operations. The interviews conducted as part of this project have identified two activities that significantly impact reducing the time of civil and criminal proceedings: the lengthy process of writing legal documents, especially judgments, and the continual production of summaries for the judge. Based on these considerations, in this paper, we aim at presenting a legal *writing assistant*, named CICERO[2], based on Natural Language Generation (NLG) techniques, that aims at enhancing process efficiency by supporting judges (and their assistants) in the drafting of legal judgments. Proposals to use AI to facilitate the work of regulators include decision-making assistance and the use of virtual assistants [3]. While certain measures have been taken to introduce such tools in the justice domain, our approach aims also to demonstrate the feasibility and usefulness of this tool.

The main contributions of this paper are the following:

– We provide a brief overview of the state of the art in NLG and explicitly describe the challenges specific to the Italian domain (Section 2).
– We introduce the pipeline that we have specifically designed for the Italian language (Section 3). The pipeline takes as input a series of Italian legal judgments, applies pre-processing techniques to extract and clean relevant information, and uses this refined data to produce a novel model, specifically tailored for our writing assistant but applicable also in more general legal domains. As we will discuss in the following, this model is valuable per se, as very few *open* language models exists for the Italian language.
– We perform both quantitative and qualitative evaluations of the model's outputs on a wide range of natural language understanding and generation tasks for Italian legal judgments, including real users. Our results demonstrate consistent improvements over the current state-of-the-art (Section 4).
– We provide an analysis and discussion of the results, validating our contributions, but more generally discussing whether it makes sense to work on specialized language models in an era dominated by commercial Large Language Models (LLMs) and chatbots, such as OpenAI ChatGPT. Our findings aims at inspiring the research community to further improve our approach (Section 5).

[*] Corresponding Author. Email: deluzi@diag.uniroma1.it
[1] Giustizia AGILE (Agile Justice in English): Innovation and efficiency in judicial offices is a research project that involves a network of 11 universities and over a hundred researchers.
[2] Cicero, the well-known politician and writer in the ancient Rome, was also a lawyer appreciated for his eloquence.
[3] Cf. https://commission.europa.eu/business-economy-euro/economic-recovery/recovery-and-resilience-facility_en

Finally Section 6 concludes the paper by drawing future work. Notably, all the artefacts produced in this work, i.e., the novel language model, the writing assistant, the outcomes of the validation (questionnaires, results, etc.) are publicly available for repeatability and according to an open science approach, cf. https://github.com/DIAG-Sapienza-BPM-Smart-Spaces/Cicero).

2 Related work

AI has made significant advancements in Natural Language Processing (NLP), with Language Modeling (LM) standing out as a particularly successful subfield demonstrating impressive and continuously improving performance across various tasks, in particular, they relevantly improved the quality of the text generated by novel NLG pipelines [31]. These recent advancements in language models have significantly enhanced the functionality and effectiveness of *writing assistants* [8]. Practical improvements have led to the development of in-depth language models, with notable examples including Bidirectional Encoder Representations from Transformers (BERT) [11] and the Generative Pretrained Transformer (GPT) series. The main differences between BERT and GPT lie in their architecture and intended use cases.

BERT is a bidirectional model that employs masked language modeling (MLM) [11] during pre-training, which allows it to generate context-sensitive embeddings between words in a sentence. It is designed for a wide range of natural language processing tasks. Conversely, GPT is a generative model that employs autoregressive language modeling (ALM) [30] during pre-training, and it is designed for tasks that require generating natural language text, such as translation, completion, and synthesis, by predicting the next word in a sequence based on the previous words.

There is a considerable difference between the masked language modeling task used in pre-training and the downstream tasks that BERT is typically used for. As a result, BERT is rarely used for text generation without fine-tuning, while GPT is more appropriate for few-shot or zero-shot text generation [4], which is one of the reasons why it was adopted in this work. Since the objective of natural language generation is to produce coherent text, decoder-based models such as GPT-1 [24], GPT-2 [25], and GPT-3 [5], which have been trained on a large corpus of text, are more adaptable and proficient in generating natural language [5].

At a general level, state-of-the-art language models can achieve excellent results using combinations of large-scale datasets. However, when trained on specific domains, the system's capabilities can lead to much more stable results at that level. Given the enormous potential for application, it is important that the legal domain, with its known language specificities and challenges, manages to rise to the challenge offered by these models, which are still relatively untested. Examples of pre-trained language models based on transformers in the legal domain are: *ALeaseBERT* [16] which was pre-trained and fine-tuned on lease data and *LegalBERT* [7], which was built by a pre-training BERT model on several legal corpora. We also note that in the legal domain, BERT is considered the current state-of-the-art for various NLP tasks, while limited research has been conducted on text generation in the same area using GPT-based models [14].

2.1 Challenges for Italian text generation

Despite the promising results shown by the models described above, their practical implementation must consider several critical aspects together with the many challenges that still require attention.

- Firstly, language models, despite their varying levels of performance in natural language processing, are trained on huge amounts of *data* and therefore require significant *computational resources*. Additionally, the implementation of these technologies, largely based on open-source software, also requires the availability of specific textual datasets (e.g., annotated corpora), lexical and semantic resources, as well as specialized skills necessary to manage training and adaptation processes for the different application domains (e.g., health, justice, finance).
- Secondly, consideration must be given to the *language* used, as these models are pre-trained mostly or exclusively in English and not in a multilingual setting. To the best of our knowledge, in the specific case of the Italian language, the set of pre-trained linguistic models publicly available is very small [21]. In particular, as regards the pre-trained language models available for text generation tasks and built on the GPT-2 architecture, there is currently only one, namely *GePpeTto* [9]. The situation changes when alternative architectures are taken into consideration. For instance, there exist various pre-trained linguistic models that are based on the BERT and RoBERTa architectures. Among these, for example, two BERT-based models: *AlBERTo* [23], an Italian model based on a slightly modified BERT architecture and trained on Twitter data, and *ITALIAN-LEGAL-BERT* [18], a pre-trained language model on a large corpus of Italian civil cases; two RoBERTa-based models, *GilBERTo* [6] and *UmBERTo* [7], both trained on large Italian corpora. However, these architectures are not appropriate for the text generation task.
- A third type of criticality is related to the fact that the pre-training datasets may contain *human biases* and *stereotypes* that are then propagated in the language model (e.g., gender bias). If approached from a technological perspective, bias can be viewed as an assessment error that undermines the accuracy and reliability of analytical results. However, from a legal perspective, bias can represent a preconceived notion that may lead to discriminatory decision-making. This raises the question of whether AI is an enabler of discrimination or whether it can be utilized to support decisions with an anti-discriminatory function, particularly at the algorithm design stage, where characteristics of the algorithm and dataset selection are considered.

To overcome these challenges, it is necessary to make available specific datasets consisting of Italian language resources not included in general archives available online, but rather related to the legal domain or generally to the language of legal texts. In this regard, new datasets for the Italian language have been recently introduced, including those gathered from Italian news websites [15] and Wikipedia articles [6]. However, there is still a lack of something

[4] Few-shot text generation uses a small number of training examples to teach a language model how to generate text, while zero-shot text generation relies on generic training data to generate text for specific tasks without the need for additional training.

[5] GPT-1 was released in 2018 with 117 million parameters and is the first pre-trained language model that is based on the autoregressive language model. In 2019, GPT-2 was released with an even larger size, featuring 1.5 billion parameters, and was capable of producing more coherent text by using a combination of techniques such as training data size and a more sophisticated attention mechanism. GPT-3, released in 2020, has a capacity of 175 billion parameters and was able to generate high-quality text suitable for various applications, including chatbots, virtual assistants, and more.

[6] Cf. https://github.com/idb-ita/GilBERTo

[7] Cf. https://github.com/musixmatchresearch/umberto

specific to the legal domain. The linguistic issue remains the biggest challenge thus far, although there have been recent advancements in adapting GPT-2 to other languages using transfer learning methods, without unnecessary retraining [10]. In the following, we provide our contribution wrt. the above scenarios.

3 CICERO

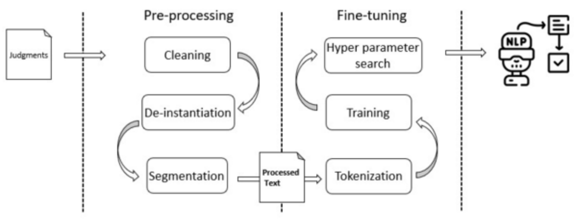

Figure 1. The CICERO pipeline.

We introduce here CICERO, our writing assistant designed to complete the sentence given an initial input text. The architecture uses GPT-2 for text generation. Specifically, the input layer receives an initial prompt for the sentence as a starting point, after undergoing cleaning, de-istantiation, and segmentation steps. In the subsequent layer, the GPT-2 language model processes the input sentence and generates a continuation of the sentence using tokenization, training, and hyperparameter search. Finally, the output layer provides the generated text, which can be displayed to the user or fed into another downstream model for further processing. The processing pipeline is shown in Figure 1.

Con ricorso del

Con il ricorso del <|DATE|> (procedimento di risarcimento ex <|LAW|>) il giudice istruttore, previa sospensione dell'esecuzione dell'esecuzione forzata e/o la declaratoria di inefficacia del provvedimento giudiziale, emetteva sentenza non definitiva per omessa indicazione in ordine alle condizioni esecutive da applicare. <|LAW|> <|NUMANN|> pubbl. <|DATE|> RG <|NUMANN|>

The above example shows an initial input text such as *"Con ricorso del"* followed by the automatically generated text in the box below with de-instantiated positions, dates and people. During pre-processing, NER techniques are used to replace named entities, such as people, places, and laws, with the names of their respective classes. De-instantiation (which will be detailed in Section 3.3) is crucial in the context of a sentence template writing assistant as simplifies customization for the user while preserving the basic sentence structure and ensuring grammatical and syntactic consistency in the generated sentences.

Below we introduce *(i)* the background knowledge required for contextualizing our work, *(ii)* the dataset used for training the model, *(iii)* its pre-processing, *(iv)* the methodology chosen for fine-tuning the model, and *(v)* the one adopted for generating text.

3.1 Background

Before delving deeper into the pre-processing and fine-tuning of the model, the main concepts have to be introduced to contextualize the work, namely *(i)* text generation, *(ii)* language modeling, and *(iii)* transformers.

Text generation

Natural language generation (NLG), which is also referred to as text generation, is one of the main subfields within natural language processing (NLP). The primary aim of text generation is to create an end-to-end solution with minimal human intervention by automatically learning an input-to-output mapping from the data [17]. In essence, natural language generation (NLG) involves the process of finding the optimal sequence of words or tokens that can be generated based on a given source sequence of information. This is typically achieved by determining the conditional probability of each token in the generated sequence, given the previous tokens and the source sequence. The ultimate goal is to find the sequence that maximizes the probability of generating the desired output. Mathematically, the search for the optimal sequence of tokens can be described as reported in [13]:

$$y_{<T+1} = \underset{y_{<T+1} \in Y}{\arg\max} \log P_\theta(y_{<T+1} \mid x)$$

Here, $y_{<T+1}$ is the output optimal sequence of length T, x is the input sequence, Y represents the set of all possible sequences, and $\log P_\theta(y_{<T+1} \mid x)$ is the logarithmic conditional probability (with parameters θ) of the sequence of T tokens based on the source sequence x. Nevertheless, it must be noticed that this is just one of the multiple strategies that can be adopted by generation.

Language modeling

Given the importance of tokens' conditional distribution in text generation, the (pre-trained) language model has acquired a central role in NLG in the last decade [17]. A language model is a statistical model whose objective is to learn a function to estimate the probability of a particular sequence of tokens. In particular, let D be a vocabulary and Y be a sequence of words $(y_1, y_2, ..., y_n)$, the objective of the language model is to learn to estimate $P(Y)$ for any sequences of y_i belonging to D:

$$P(Y) = P(y_1, y_2, ..., y_n) = \prod_{i=1}^{n} P(y_i \mid y_1 ... y_{n-1})$$

After the training of a language model is completed, the learned probability distributions can be used directly for text generation, which involves generating the next word in a sentence based on the probabilities estimated by the language model [25]. Alternatively, the language model can be fine-tuned on a small dataset to adapt it for a specific supervised task. The latter is the main idea behind the usage of the pre-trained language model (PLM) [17].

Transformers

The paradigm of NLG has been deeply influenced by encoder-decoder models [13]: the encoder maps the input sequence into fixed-size low-dimensional vectors, known as input embeddings; the decoder then generates a target text based on these embeddings. Differently from earlier statistical approaches, which relied on explicit feature engineering, encoder-decoder architecture is able to extract the important features automatically during training [17]. This makes it easier to capture complex relationships between inputs and outputs, resulting in better performance on text generation tasks.

Among the various encoder-decoder architectures, transformers [28] have proven to be particularly successful in learning precise latent feature representations for language modeling [13]. Both the encoder and decoder of the original transformer can be described as composed by the repetition of identical blocks. The encoder's block has a multi-head self-attention module and a position-wise feed-forward network [19]. The encoder has the task of mapping the input to a latent representation of the whole sequence [28]. The decoder block has cross-attention modules added between the multi-head self-attention modules and the position-wise FFNs [19]. Its task is to transform the latent representation in the desired target [19].

Inter alia, GPT2 is a transformer-based model that is built using solely transformer decoder blocks [25]. This model has shown to be particularly fit for natural language generation [17].

3.2 Dataset

In this work, we consider a set of 37 000 legal documents, including associated metadata, provided by the Italian Ministry of Justice. Approximately 92% of the documents are court decisions, while the remaining documents are ancillary acts of the courts. The original source documents were organized into folders according to the district (i.e., geographical area) of the Italian legal system, with each folder containing CSV files. Within each CSV file, a single row housed both the metadata and text of a single document.

3.3 Pre-processing and de-instantiation

In the pre-processing phase, the initial documents are refined in such a way that they can be used for fine-tuning the model [1]. The pre-processing of civil judgments requires overcoming several challenges derived from specific features of these documents. Indeed, judgments are characterized both by a semi-structured frame and by a special vocabulary, related to the legal lexicon and uncommon in ordinary Italian.

The ad-hoc pre-processing pipeline implemented in this work involves several steps, including *(i)* cleaning the document, *(ii)* de-instantiating its named entities, and *(iii)* segmenting the documents into smaller chunks. The goal of this phase is to extract meaningful data from the raw text input and prepare it for the fine-tuning phase. Once the pre-processing phase is completed, the resulting corpus of pre-processed Italian sentences is used to fine-tune the pre-trained language model (GPT-2). This process enables the language model to learn the nuances and intricacies of Italian legal text and generate plausible text.

Firstly, data cleaning involves removing any irrelevant information from the textual data and standardizing the corpus with regard to Italian syntax. For instance, the indicators and the structural elements (required to store the legal judgments in the database) are erased to organize the text with a sequential structure. Therefore, text cleaning did not affect in a relevant way the content or the style of the judgments, it just made it accessible for a language model. The reason behind this choice is that the final aim of text generation is to return a new text sample that replicates the original style of the text. Therefore, extensive text cleaning may eliminate important details that should have been learned by the model.

Secondly, the document was de-instantiated. This is a novel data pre-processing technique that was added to our pipeline, based on NER[8]. The core concept of de-instantiation is to pre-process the text so that named entities, such as people, locations, and laws would have been substituted with the name of their classes [12]. Therefore, the language model trained on the de-instantiated documents will learn to return those special tokens during text generation, rather than particular instances (e.g., a law, a proper name, or a location). This is

[8] Actually, one could interpret de-instantiation as a type of masking[2], where, rather than using an anonymous mask, a semi-anonymous NER token is deployed.

particularly useful in production when the model is used as a writing assistant for legal users. Indeed, the model should generate general sentences that can be instantiated by the legal user with the particular details of his/her case. Otherwise, the user will have to manually erase every particular information; this would make meaningless the usage of the writing assistant. With regard to the technical details of the de-instantiation phase, a pre-trained NER model from Spacy[9] was used to label the entities in the text. Once the entities were labeled, they have been substituted with a special token based on their labels (such as <|`LAW`|>). The Spacy model was already fine-tuned on the legal domain and downloaded from Hugging Face[10]. An example is reported below.

Nella causa civile

n.r.g. 33/2015, tenutasi il 23/02/2015 promossa da: ROSSI MARIO (C.F:RSSMRAI84M22A100A) elettivamente domiciliato in Pesaro [...].

Nella causa civile

n.r.g. NUMERO/ANNO, tenutasi il DATA promossa da: PERSONA (C.F:CODICEFISCALE) elettivamente domiciliato in LUOGO [...].

Lastly, the documents have been segmented into smaller chunks. This is required for two reasons. Firstly, it allowed us to eliminate useless details regarding the structure and the division of the judgment. Secondly, reducing the dimension of the documents was required due to the polynomial scalability of the memory of Large Language Models over the input size. The first step of this phase was to divide the judgment into chapters. Indeed, some of the chapters of a judgment do not contain any useful information for the model but they just present some formal information about the judges, the tribunal, and the parts of the process. Once we extracted the searched chapters, we segmented them into sentences through a Spacy Model. Thereafter, a new corpus of smaller sentences was built and the initial corpus of judgments was discarded.

3.4 Fine-tuning

The fine-tuning phase consists of several steps, including tokenization, model training, and hyperparameter search. The starting point was to select the language model that had to be fine-tuned. For this stage, it was used an Nvidia RTX 3090 with 24GB of VRAM. Several attempts were made to choose the models, trying different alternatives based on GPT-2 [25]. Nevertheless, as previously mentioned, the availability of Italian LLMs is extremely constrained. A small Italian language model with 124M parameters based on a GPT-2 architecture was chosen. Specifically, the language model was already adapted to Italian through a novel technique based on retraining lexical embeddings to gain embeddings for Italian that are aligned with GPT-2's original lexical embeddings. Therefore, the starting point was the `GroNLP/gpt2-small-italian` [10] checkpoint downloaded through Hugging Face[11]. The smaller version was preferred to the medium one, `GroNLP/gpt2-medium-italian-embeddings`[12]. Indeed, already in the original paper, the small version of the model

[9] Cf. https://spacy.io
[10] Cf. https://huggingface.co/bullmount/it_nerIta_trf
[11] Cf. https://huggingface.co/GroNLP/gpt2-small-italian
[12] Cf. https://huggingface.co/GroNLP/gpt2-medium-italian-embeddings

performed better than medium one [10][13]. The authors justify this counterintuitive result owing to an incomplete training of the model of medium size [10]. Furthermore, this aforementioned version of GPT-2, only adapted to Italian, performed better than *GePpeTo* [9], a small version of GPT-2 (117M parameters) trained from scratch for Italian. Instead, GePpeTo has been used as the baseline.

The first step in fine-tuning the model was tokenization. This involves breaking down the raw text data into smaller units, known as tokens, which can be fed into the model. Tokenization is critical for creating a numerical representation of the text data that can be understood by the model. After tokenization, the model is trained using the tokenized data. During training, the model learns to recognize patterns in the input data and make accurate predictions based on those patterns [22]. The goal of training is to optimize the model's parameters in such a way that the likelihood of the training data is maximized, without reaching overfitting. Indeed, the language model should learn to estimate the probability of a word, given the previous words.

Once a probabilistic model is trained, the model can be used for autoregressive sampling: starting from one or more words, it estimates the probability distribution of the next word and samples from it. This allows the model to generate text from an initial prompt. In practice, we trained the model for 3 epochs over the pre-processed data. We set the learning rate to 2e-5 and the weight decay to 0.01. The learning rate controls the step size of the gradient descent optimizer during training, while the weight decay helps to prevent overfitting by regularizing the weights of the model. The batch size was set to 64. Furthermore, we implemented gradient accumulation and checkpointing. Gradient accumulation involves accumulating the gradients of multiple small batches before updating the weights of the model, effectively simulating a larger batch size. Checkpointing involves periodically saving the state of the model during training to reduce memory usage and prevent out-of-memory errors. The complete list of hyperparameters is shown in Table 1. We remind that all the code used for the implementations is available at the link https://github.com/DIAG-Sapienza-BPM-Smart-Spaces/Cicero).

Table 1. Hyperparameters list.

Parameter	Value
evaluation_strategy	"epoch"
learning_rate	2e-5
weight_decay	0.01
num_train_epochs	3
per_device train_batch_size	64
per_device_eval_batch_size	64
gradient_accumulation_steps	8
gradient_checkpointing	True
fp16	True

4 Evaluation

To assess the performance of the models, two types of evaluation were conducted: an automatic evaluation based on the perplexity of the LLMs, and a human evaluation carried out by a group of users specialized in law and legal documents.

4.1 Automatic evaluation

The quantitative evaluation aimed at demonstrating the efficacy of fine-tuning models on specific fields, such as the Italian legal language. Consequently, it was decided to confront the original models with their fine-tuned counterparts computed by us. To compare the models we measured *perplexity*, a commonly used metric for evaluating the quality of a language model that measures how well the language model can predict the next word in a sequence of text. In particular, the quality of the models' predictions is operationalized in terms of the "predictability" of the latter: a language model is assumed to be good if it is able to predict with high accuracy the tokens that will be seen next [4]. Specifically, the mathematical formulation of perplexity derives from information theory [4]. Here, the perplexity function $PPL(\cdot)$ was defined to measure the performance of probability distributions or models in predicting a given sample, relative to the actual observed data or ground truth [17]. A lower perplexity score indicates that the probability distribution or model is more accurate in predicting the sample, and thus better at approximating the underlying probability distribution of the data [4]. Therefore, the objective of training a language model is to minimize the value of $PPL(P(y))$. In particular, the perplexity of a discrete probability distribution $P(y)$ is defined as [17]:

$$PPL(P(y)) := e^{H(P(y))} = e^{-\sum_y P(y)\log P(y)} = \prod_y P(y)^{-P(y)}$$

where $H(P(y))$ is the entropy of the distribution $P(\cdot)$.

Once the two versions of GroNLP were fine-tuned following the procedure previously described, perplexity was implemented and computed to select one of the models for two purposes: *(i)* serving as the backbone of the writing assistant, and *(ii)* conducting subsequent qualitative evaluation. Regarding the implementation of perplexity, the metrics were based on the official script proposed by Hugging Face[14].

In terms of results, the fine-tuned version of GroNLP demonstrated significantly improved performance for both the medium and small models, underscoring the importance of the training process. `cicero-gpt2` (i.e., the fine-tuned model derived from `GroNLP/gpt2-small-italian`) outperformed the `cicero-gpt2-medium` (i.e., the fine-tuned model derived from `GroNLP/gpt2-medium-italian-embeddings`). Consequently, the `cicero` model was selected for deployment in the writing assistant CICERO. Results are shown in Table 2.

Table 2. Perplexity.

Model	Perplexity
`GroNLP/gpt2-small-italian`	46.2063
`cicero-gpt2`	18.7136
`GroNLP/gpt2-medium-italian-embeddings`	46.4573
`cicero-gpt2-medium`	25.9911

4.2 Human evaluation

To carry out and report the human evaluation in NLG, the guidelines in [27] were followed (see Figure 2). The first step is determining the goal of the evaluation, as both the type of evaluation and the type of

[13] The `GroNLP/gpt2-medium-italian-embeddings` was fine-tuned using an Nvidia DGX system with eight A100 GPUs with 40GB of VRAM in parallel.

[14] Cf. https://huggingface.co/docs/transformers/perplexity

research depends on it. Since the objective is to collect human feedback on the quality of the generated output, with a particular focus on text characteristics, it is an *intrinsic* evaluation of a *quantitative* nature. For quantitative assessments, the constructs of interest must be defined, as part of the next step of the process. Among them, the type of questions was chosen to emphasize the fact that the model produces texts that readers evaluate as high quality, namely *impact questions*. For the *criteria*, separate criteria should be used, as suggested by [27]. We follow this recommendation, further refining the choice based on the legal context (for example, we chose legal lexicon as a specific criterion, in addition to other more general criteria on text quality). We also provide a definition of the criteria and concrete examples in the instructions before the questionnaire. This approach assists participants in gaining a clearer comprehension of the criteria, thereby mitigating any confusion that may compromise the validity of the assessment. Choosing an appropriate rating scale and size is crucial: we selected a 4-point Likert scale, which is widely used in NLG assessment, and provided an interpretation of the scale before the questions. We selected the *sample* to reflect the large-scale target audience [3], and recruited 27 legal domain experts, including judges, lawyers, Officers for the Process (UPP), PhD students, as well as undergraduate and graduate students in Law.

In the study *design* phase, we chose to use a pre-developed online survey software (Google Forms), which is currently the most commonly used method and allows for customization of various aspects and the survey layout (web design). The questionnaire was completed online under our supervision, either in person or through an online meeting where participants shared their screens. To minimize potential distortion effects, we carefully considered the appearance of the survey by presenting the questionnaire on two pages instead of one long scrolling page, displaying the questions in random order, using a predefined Google Forms function, and attempting to create a simple and engaging survey for participants.

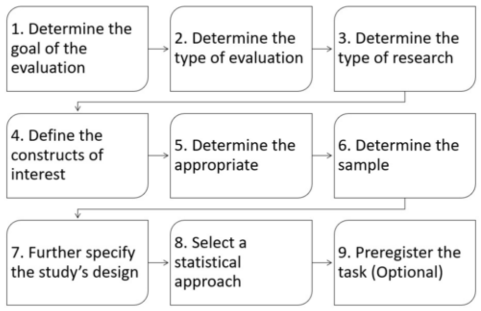
1. Determine the goal of the evaluation
2. Determine the type of evaluation
3. Determine the type of research
4. Define the constructs of interest
5. Determine the appropriate
6. Determine the sample
7. Further specify the study's design
8. Select a statistical approach
9. Preregister the task (Optional)

Figure 2. List of stages and steps for conducting a human evaluation of automatically generated text as suggested in [27].

Evaluated language models. Regarding human evaluation, it was deemed appropriate to compare distinct models from those employed in the quantitative evaluation. The decision was based on the observation that the objectives of the two evaluations were marginally divergent. While the quantitative evaluation aimed at demonstrating the efficacy of fine-tuning, the objective of the human evaluation was to evaluate the potential future benefits of a legal writing assistant. Therefore, we selected models that were considerably different from each other to get a more comprehensive overview. Specifically, we have chosen the following models for evaluation:

- *GePpeTto* [9], the first language model trained solely on a collection of Italian texts. In this evaluation, GePpeTo was used as a baseline;
- CICERO, our fine-tuned model and assistant;
- *ChatGPT*[15], the language model and chatbot developed by OpenAI. It can be considered today as a gold standard text generator.

Sentences and structure of the questionnaires. For the questionnaire, we generated sixty sentences. However, we were concerned that a lengthy questionnaire could overwhelm participants with an excessive number of sentences to evaluate, leading to superficial ratings due to boredom or fatigue. For this reason, we decided to randomly select twenty sentences to be evaluated out of the sixty produced for each participant. This allowed us to evaluate a more heterogeneous set of sentences while requiring each participant to evaluate only a subset of the total sentences. Subsequently, the results from all forms were combined and analyzed as a single dataset.

The questionnaire comprises two distinct sections, namely:

1. *Sentence evaluation*. This section features eight sentences, with two derived from legal judgments and two obtained from each of the three chosen language models. For any phrases, participants rate five text characteristics (Grammar, Juridical Lexicon, General Meaning, Ripetitivity, and Sentence Syntax) by casting a grade from one to four [16];
2. *"Real" versus "Artificial"*. This section includes twelve phrases, three sourced from legal judgments and three obtained from language models. Participants are informed that any sentence may either be "Real", extracted from a civil judgment, or "Artificial", generated automatically by an NLG model, and then are required to categorize any phrase accordingly.

Ensuring impartiality in the entire evaluation process was a crucial step. Therefore, to prevent any potential bias by the paper's authors, the sentence generation phase was achieved through a systematic and rigorous approach. Initially, appropriate inputs were selected. Sentences extracted from civil judgments were randomly chosen from those that commenced with the selected inputs [17]. Similarly, sentences generated by the models were produced starting from the same incipit. For each model and input, five possible sentence completions were generated, and the best completion was selected through a vote by the research team members; to avoid any bias, the team members were not informed of each other's choices.

5 Results and discussion

In this section, we will show the results of our human evaluation. First, we will present CICERO's performance against GePpeTto and real judgments, then ChatGPT's performance compared to CICERO and real judgments. A total of 27 participants replied to the questionnaire (results are shared via the link provided in Section 1).

[15] https://openai.com/blog/chatgpt

[16] Regarding text characteristics: *Grammar* assesses whether the text is error-free, *Juridical Lexicon* evaluates whether the vocabulary used is congruent with legal vocabulary, *General Meaning* assesses whether the sentence makes logical sense, *Repetitivity* indicates whether there are no redundant elements, and *Sentence Syntax* indicates whether the parts of the text are well connected. About the scale: *1*-Very bad, *2*-Bad *3*-Good *4*-Very good.

[17] Please note that sensitive data (e.g. names, places, and dates) have been replaced by fictitious data.

Results CICERO - GePpeTto - Legal judgments. Overall, the results of the *Sentence Evaluation* section of the questionnaire, show that the best-rated sentences (indicated in terms of averages on 4-point Likert scale) are that of legal judgments (3.41), followed by CICERO (3.04) and then GePpeTto (2.54), but the latter is much further away from legal judgments than our model (Table 3). If we look at the results in more detail, it will be noted that the judgments get the highest preference (3.41), despite the fact that the participants highlighted a key feature, namely the fact that the real text often had errors.

Table 3. CICERO, GePpeTto, Legal Judgment Sentence Evaluation results.

	Cicero	GePpeTto	Legal Judgment
Grammar	2,91	2,52	3,35
Juridical Lexicon	3,04	2,44	3,31
General Meaning	2,83	2,3	3,56
Ripetitivity	3,48	3,09	3,56
Sentence Syntax	2,93	2,37	3,28
TOTAL AVERAGE	3,04	2,54	3,41

As for the *"Real" versus "Artificial"* section, the results show that the participants mostly recognized the sentences generated by GePpeTto as artificial, in 68% of cases, whereas they were able to recognize the real sentences with an even higher percentage. However, we expected higher values for the latter, meaning that the expert was not able to identify whether the text was written by a human or a machine. The interesting finding pertains to CICERO, as about 50% of the participants were unable to distinguish between the real and the generated sentences. See the results in Table 4.

Table 4. CICERO, GePpeTto, Legal Judgment *"Real" versus "Artificial"* results.

	Cicero	GePpeTto	Legal Judgment
Real	52%	32%	78%
Artificial	48%	68%	22%

Results CICERO - ChatGPT - Legal Judgment. Here we highlight further results regarding human evaluation. In particular, as can be seen in Tables 5 and 6, ChatGPT, which was chosen as the gold standard for text generation, exhibited better ratings than CICERO and even sentences extracted from legal judgments. Precisely, in both sections, ChatGPT achieved the best results. In the *Sentence Evaluation* section, ChatGPT's overall average was 3.70 out of 4. This result, in addition to being the best one (the second best was 3.41, recorded by legal judgment) is enhanced by the fact that OpenAI's model achieved the highest score in each of the five evaluated textual characteristics highlighting ChatGPT's ability to adapt to specific tasks [18]. The most interesting result, though, was the *"Real" versus "Artificial"* section: participants believed that 88% of ChatGPT's sentences were being extracted from civil judgments, which is exactly 10% higher than the percentage recorded by sentences extracted from civil judgments (78%). Next, we will argue how these results contribute to demonstrating the convenience of introducing supporting tools to assist users in the task of writing legal documents. However, before proceeding further, it is necessary to make some remarks regarding ChatGPT limitations.

[18] To obtain optimal outputs during the execution of specific tasks from general-purpose LLMs such as ChatGPT, it is necessary to properly construct prompts containing precise instructions. This training phase is referred to as prompt engineering [29].

Discussing ChatGPT. Among the characteristics of ChatGPT, the most interesting for the subsequent observations is that it is a closed-source model. Closed-source models do not allow users to modify or even access the underlying source code, resulting in an inability to interpret data and a lack of model transparency. In addition, responsibility issues may arise if the model produces inaccurate or discriminatory results, which are current challenges. If the purpose of this paper is to lay the foundations for the development of a writing assistant to be used in Italian e-justice, it is difficult to assume that a public administration might rely on a closed-source artefact.

Another aspect not to be underestimated is that ChatGPT's APIs, which are essential for its eventual integration within a writing assistance service, are fee-based. The expensive licenses required could make it impractical for nationanwide adoption.

Table 5. CICERO, ChatGPT, Legal Judgment Sentence Evaluation results.

	Cicero	ChatGPT	Legal Judgment
Grammar	2,91	3,72	3,35
Juridical Lexicon	3,04	3,61	3,31
General Meaning	2,83	3,74	3,56
Ripetitivity	3,48	3,8	3,56
Sentence Syntax	2,93	3,65	3,28
TOTAL AVERAGE	3,04	3,7	3,41

Table 6. CICERO, ChatGPT, Legal Judgment *"Real" versus "Artificial"* results.

	Cicero	ChatGPT	Legal Judgment
Real	52%	88%	78%
Artificial	48%	12%	22%

6 Concluding remarks

Our research is motivated by devising a supporting tool to speed up the process of writing legal judgments. The need for such a writing assistant was highlighted by a domain study and reinforced by human evaluations, which revealed that handwritten sentences are prone to errors. Results obtained by ChatGPT in the human evaluation confirm that the current state-of-the-art of language models could address this proven need in the near future. However, the closed-source nature of ChatGPT poses some scientific research challenges. Thus, we have demonstrated that an open-source writing assistant like CICERO would be ideal. CICERO appears to be able to fulfil all the presented desiderata, showing satisfactory results and a margin for improvement. CICERO has already demonstrated: *(i)* that it is beneficial to use fine-tuning for specific application domains, in the quantitative evaluation, and *(ii)* can achieve considerably better results than the baseline of models suitable for generating Italian text (namely GePpeTto), as shown in the human evaluation. Regarding possible improvements, CICERO is based on a GPT2 architecture. It is easily conceivable that improvements in the model's performance could be obtained by applying the training pipeline we have presented in this paper to new, more performant architectures in the Italian language domain.

Concluding, we tried to answer the question of whether it does make sense to implement a specialized assistant in the LLM era. The quantitative results show that fine-tuning of models was successful, obtaining better perplexity values than the currently available baseline(s). On the other hand, human evaluation produced unexpected results: ChatGPT outperformed the original legal judgments in every section. This proves that there is a large room for improvement for *specialized open-source platforms* such as ours. Our future work will be towards improving our model and trying to match ChatGPT performances, in particular with bigger datasets and deeper models.

Acknowledgements

This work has been partly supported by the Italian Ministry of Justice PON project "Per una Giustizia giusta: Innovazione ed efficienza negli uffici giudiziari - Giustizia Agile", and by the PNRR-PE-AI "FAIR" project funded by the NextGenerationEU program. The authors would like to thank research fellows who actively participated in the validation of the tool within the Giustizia Agile project, contributing their valuable insights to the successful completion of this study.

References

[1] Murugan Anandarajan, Chelsey Hill, Thomas Nolan, Murugan Anandarajan, Chelsey Hill, and Thomas Nolan, 'Text preprocessing', *Practical text analytics: Maximizing the value of text data*, 45–59, (2019).

[2] Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, et al., 'Unilmv2: Pseudo-masked language models for unified language model pre-training', in *Int. conf. on machine learning*, pp. 642–652. PMLR, (2020).

[3] Anja Belz and Ehud Reiter, 'Comparing Automatic and Human Evaluation of NLG Systems', in *EACL 2006, 11st Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference,2006*, eds., Diana McCarthy and Shuly Wintner. The Association for Computer Linguistics, (2006).

[4] Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Jennifer C. Lai, and Robert L. Mercer, 'An Estimate of an Upper Bound for the Entropy of English', *Computational Linguistics*, **18**(1), 31–40, (1992).

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al., 'Language models are few-shot learners', *Advances in neural information processing systems*, **33**, 1877–1901, (2020).

[6] Silvia Casola and Alberto Lavelli, 'WITS: Wikipedia for Italian Text Summarization', in *Proceedings of the Eighth Italian Conference on Computational Linguistics, CLiC-it 2021*, eds., Elisabetta Fersini, Marco Passarotti, and Viviana Patti, volume 3033 of *CEUR Workshop Proceedings*. CEUR-WS.org, (2021).

[7] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. LEGAL-BERT: The muppets straight out of law school. arXiv preprint arXiv:2010.02559, 2020.

[8] Council of Europe, 'European judicial systems: Efficiency and quality of justice', *CEPEJ Studies*, **26**, 1–338, (2018).

[9] Lorenzo De Mattei, Michele Cafagna, Felice Dell'Orletta, Malvina Nissim, and Marco Guerini, 'GePpeTto Carves Italian into a Language Model', in *7th Italian Conf. on Computational Linguistics, CLiC-it 2020*, eds., Johanna Monti, Felice Dell'Orletta, and Fabio Tamburini, volume 2769, (2020).

[10] Wietse de Vries and Malvina Nissim. As good as new. How to successfully recycle English GPT-2 to make models for other languages, 2020.

[11] 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, NAACL-HLT 2019*, eds., Jill Burstein, Christy Doran, and Thamar Solorio, pp. 4171–4186. Association for Computational Linguistics, (2019).

[12] Beniamino Di Martino, Fiammetta Marulli, Pietro Lupi, and Alessandra Cataldi, 'A machine learning based methodology for automatic annotation and anonymisation of privacy-related items in textual documents for justice domain', in *14th Int. Conf. on Complex, Intelligent and Software Intensive Systems (CISIS)*, pp. 530–539. Springer, (2021).

[13] Chenhe Dong, Yinghui Li, Haifan Gong, Miaoxin Chen, Junxin Li, Ying Shen, and Min Yang, 'A Survey of Natural Language Generation', *ACM Comput. Surv.*, **55**(8), (2022).

[14] Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. Ammus: A survey of transformer-based pretrained models in natural language processing. arXiv preprint arXiv:2108.05542, 2021.

[15] Nicola Landro, Ignazio Gallo, Riccardo La Grassa, and Edoardo Federici, 'Two New Datasets for Italian-Language Abstractive Text Summarization', *Information*, **13**(5), 228, (2022).

[16] Spyretta Leivaditi, Julien Rossi, and Evangelos Kanoulas. A benchmark for lease contract review. arXiv preprint arXiv:2010.10386, 2020.

[17] Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. Pretrained Language Models for Text Generation: A Survey, 2022.

[18] Daniele Licari and Giovanni Comandé, 'ITALIAN-LEGAL-BERT: A pre-trained transformer language model for italian law', in *Companion Proceedings of the 23rd International Conference on Knowledge Engineering and Knowledge Management, Bozen-Bolzano, Italy, September 26-29, 2022*, volume 3256 of *CEUR Workshop Proceedings*. CEUR-WS.org, (2022).

[19] Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. A Survey of Transformers, 2021.

[20] Neha Mahajan, 'E-governance: its role, importance and challenges', *Int. J. Curr Innov Res*, **1**(10), 237–243, (2015).

[21] Alessio Miaschi, Gabriele Sarti, Dominique Brunato, Felice Dell'Orletta, and Giulia Venturi, 'Italian Transformers Under the Linguistic Lens', in *Proceedings of the Seventh Italian Conference on Computational Linguistics, CLiC-it 2020*, volume 2769 of *CEUR Workshop Proceedings*. CEUR-WS.org, (2020).

[22] Lazar Peric, Stefan Mijic, Dominik Stammbach, and Elliott Ash, 'Legal language modeling with transformers', in *4th Workshop on Automated Semantic Analysis of Information in Legal Text (ASAIL 2020)*, volume 2764. CEUR-WS, (2020).

[23] Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile, 'AlBERTo: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets', in *Proceedings of the Sixth Italian Conference on Computational Linguistics, Italy, 2019*, eds., Raffaella Bernardi, Roberto Navigli, and Giovanni Semeraro, volume 2481 of *CEUR Workshop Proceedings*, (2019).

[24] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al., 'Improving language understanding by generative pre-training', (2018).

[25] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al., 'Language models are unsupervised multitask learners', *OpenAI blog*, **1**(8), 9, (2019).

[26] Oreste Signore, Franco Chesi, and Maurizio Pallotti, 'E-government: challenges and opportunities', in *CMG Italy-XIX annual conference*, volume 7, pp. 177–198, (2005).

[27] Chris van der Lee, Albert Gatt, Emiel van Miltenburg, and Emiel Krahmer, 'Human evaluation of automatically generated text: Current trends and best practice guidelines', *Comput. Speech Lang.*, **67**, 101151, (2021).

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, 'Attention is all you need', *Advances in neural information processing systems*, **30**, (2017).

[29] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C Schmidt, 'A prompt pattern catalog to enhance prompt engineering with chatgpt', *arXiv preprint arXiv:2302.11382*, (2023).

[30] Jian Yang, Shuming Ma, Dongdong Zhang, Shuangzhi Wu, Zhoujun Li, and Ming Zhou, 'Alternating Language Modeling for Cross-Lingual Pre-Training', in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 9386–9393. AAAI Press, (2020).

[31] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al., 'A Survey of Large Language Models', *arXiv preprint arXiv:2303.18223*, (2023).