

ALBERTurin: A Fully Open Family of Italian Encoder Models with Modern Architectures

Matteo Rinaldi^{1,*†}, Marco Madeddu^{1,†}, Calogero Jerik Scozzaro^{1,†}, Matteo Delsanto¹, Daniele Paolo Radicioni¹ and Viviana Patti¹

¹University of Turin, Computer Science Department, Corso Svizzera 185, 10149 Torino, Italy

Abstract

Encoder-only Transformer models remain a fundamental component of modern NLP, powering applications such as text classification, retrieval, and embedding generation. While recent work has introduced several architectural improvements to enhance efficiency and support longer contexts, these advances have primarily focused on English and multilingual settings, with comparatively limited attention devoted to Italian encoder models. In this work, we introduce ALBERTURIN, the first fully open family of modern Italian encoder-only Transformers. We first investigate different positional embedding strategies for long-context modeling and introduce GETTONE, a new tokenizer optimized for Italian. Building on these findings, we train three encoder models of different sizes that incorporate recent architectural advances. Despite being trained on substantially fewer tokens than several existing Italian encoders, our models achieve competitive downstream performance while supporting longer input contexts and significantly faster training. To promote reproducible research, we release all models, training data, code, and implementation details.

Keywords

Language Modeling, Pre-Training, Long Context, Italian Language Model, Encoder Models

1. Introduction

Transformers architectures have revolutionized Natural Language Processing (NLP) [1]. They rely on the attention mechanism [2], which enables the model to capture dependencies between tokens independently of their distance in the input sequence while allowing efficient parallel computation. The original architecture comprises two main components: an encoder, designed to produce contextualized representations of the input, and a decoder, responsible for autoregressive token generation conditioned on the encoded sequence. These two components can also be disentangled, giving rise to specialized architectures tailored to different classes of tasks. In recent years, decoder-only architectures have become the dominant paradigm in NLP thanks to their rapid scaling in both model size and capabilities, evolving from early Large Language Models (LLMs) such as GPT-2 [3] and GPT-3 [4], which were often prone to generating incoherent or unreliable text, to more recent systems [5, 6, 7] that achieve state-of-the-art performance across a wide range of tasks (e.g., question answering [8, 9], code generation [10, 11], mathematical reasoning [12, 13], and agentic tool-use capabilities [14, 15]).

Encoder architectures became widely adopted following the introduction of BERT [16] in 2018, although research attention later progressively shifted toward decoder-only models. Nevertheless, encoder architectures remain widely used for classification and embedding-based tasks and continue to see widespread adoption in practice. More recently, they have also attracted renewed interest driven by two main factors: *i*) encoder models are particularly effective at producing vector representations, making them a key component of Retrieval-Augmented Generation (RAG) systems [17], which are employed

CLiC-it 2026: Twelfth Italian Conference on Computational Linguistics, September 14–16, 2026, Palermo, Italy

*Corresponding author.

†These authors contributed equally.

✉ matteo.rinaldi@unito.it (M. Rinaldi); marco.madeddu@unito.it (M. Madeddu); calogerojerik.scozzaro@unito.it (C.J. Scozzaro); matteo.delsanto@unito.it (M. Delsanto); daniele.radicioni@unito.it (D.P. Radicioni); viviana.patti@unito.it (V. Patti)

ORCID: 0009-0004-7488-8855 (M. Rinaldi); 0009-0004-5620-0631 (M. Madeddu); 0009-0003-8311-7801 (C.J. Scozzaro); 0000-0002-7582-9185 (M. Delsanto); 0000-0003-0443-7720 (D.P. Radicioni); 0000-0001-5991-370X (V. Patti)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

to mitigate hallucinations in LLMs [18]; and *ii*) the growing interest in generative diffusion models, which investigate non-autoregressive approaches to text generation and use latent representations for generative modeling [19].

Despite the strong performance of multilingual models driven by the increasing scale of training data and model parameters, language-specific architectures are still highly relevant. Previous work has shown that monolingual models are better able to capture language-specific nuances and can achieve improved downstream performance compared to multilingual models when trained on sufficiently large corpora [20]. Although Italian is not a low-resource language, the amount of available data remains significantly smaller than the trillions of tokens typically used to train modern English-centric language models. This has motivated recent efforts to build large-scale Italian textual resources, such as TestiMole [21]. Furthermore, the increasing computational costs and engineering complexity required to train modern models have progressively shifted their development toward large industrial actors. As a consequence, many state-of-the-art models are released without access to training data and code, limiting reproducibility and making it difficult to properly assess issues such as data contamination [22]. In response to this trend, initiatives such as OLMo [23] and Apertus [24] promote fully open decoder-only language models by releasing datasets, training code, checkpoints, and evaluation pipelines. In the Italian landscape, Minerva [25] represents one of the few decoder-only language model families releasing both model weights and training data.¹ However, to the best of our knowledge, no fully open Italian encoder model has been released. Another limitation of existing Italian encoder models is the reliance on the original BERT context window of 512 tokens, limiting their applicability to long-document processing and embedding-based retrieval scenarios.

In this work, we present ALBERTURIN, a family of modern encoder models for Italian that incorporates recent architectural improvements and extends the context window. We also introduce GETTONE, a new tokenizer optimized for Italian and adopted across all models in the family. Since several approaches have been proposed to improve long-context handling, we first investigate three positional embedding strategies: absolute positional embeddings, RoPE [26], and ALiBi [27]. Based on the results of these experiments, we select the best-performing configuration and use it to train ALBERTMINI, ALBERTINA, and ALBERTONE101, comprising approximately 95M, 140M, and 450M parameters, respectively. Our models achieve competitive performance with state-of-the-art Italian encoder models despite being trained on a smaller number of tokens, while additionally supporting longer input contexts. To promote transparent and reproducible research on language models, we release all the code, training data, and evaluation pipelines used in this work.²

2. Related Work

During the 2010s, the NLP community increasingly focused on learning distributed text representations, progressing from static word embeddings [28] to contextualized representations [29]. Starting from this trend and adding the attention mechanism, Devlin et al. [16] introduced BERT, an encoder-only Transformer architecture trained with two bidirectional pre-training objectives: masked language modeling (MLM) and next sentence prediction (NSP). The authors also released a multilingual model, mBERT, trained on 104 languages, including Italian. Later, many works have tried to improve the BERT architecture with changes to the already solid starting point. Firstly, Liu et al. [30] demonstrated with the RoBERTa architecture that the original BERT was under-trained and that modifications such as removing the NSP objective and extending the training schedule lead to significant performance gains, a finding later confirmed by Samuel et al. [31]. Subsequently, Conneau et al. [32] adapted this approach to a multilingual setting by training XLM-RoBERTa (XLM-R) on more than two terabytes of filtered CommonCrawl data covering one hundred languages. Most recently, several works have developed encoder-only models with updated architectures, larger corpora, and improved training pipelines. Notable examples include: NomicBERT [33], which introduced an 8192 token context window;

¹To the best of our knowledge, the training code has not been publicly released.

²<https://huggingface.co/ALBERTurIn>

NeoBERT [34], which systematically explore a series of architectural modifications to the original BERT model; ModernBERT [35], which introduces architectural improvements and is trained on more than two trillion tokens; and EuroBERT [36], a family of multilingual encoders covering European and widely spoken global languages.

While multilingual encoder models have demonstrated strong cross-lingual generalization, several studies have shown that models trained on a single language typically outperform their multilingual counterparts on monolingual tasks [37, 38, 39]. This observation has motivated the development of Italian-specific BERT models [40]. Among the earliest and most widely adopted Italian models is the BERT-*dbmdz* series, released in both standard³ and XXL⁴ variants. These models were trained on approximately 2 billion tokens with a 512 sequence length for roughly 2–3 million steps. Two other significant monolingual models are GilBERTo and UmBERTo. GilBERTo⁵ is an uncased model based on the RoBERTa architecture and the CamemBERT [20] tokenization approach. It was trained with the MLM objective for 100k steps on the Italian Web corpus section of the OSCAR corpus [41], comprising approximately 11 billion words. UmBERTo⁶ is a RoBERTa-based cased model trained on the Italian portion of the OSCAR corpus. It employs Whole Word Masking (WWM), a masking strategy in which an entire word is masked whenever at least one of its subword tokens is selected. Finally, a lightweight alternative is BERTino [42], a DistilBERT-based [43] model trained on two large general-domain Italian corpora: PAISA [44] and ItWaC [45]. In the domain-specific setting, two Italian BERT models have been developed. AlBERTo [46] is a BERT model trained on 200 million tweets from TWITA [47], a large corpus of Italian tweets, using solely the MLM objective and a maximum context window of 128 tokens. BERToldo [48] is an Italian adaptation of a historical BERT model, trained on LiberLiber,⁷ a collection of over 4,000 Italian books, and on Wikisource,⁸ an online digital library maintained by Wikimedia.

Despite the availability of several natively Italian encoder models, three important limitations remain: *i*) existing models rely on a maximum context window of 512 tokens, which restricts their applicability to tasks involving long documents; *ii*) documentation is often sparse, and in some cases no accompanying paper is available, making reproducibility and comparative evaluation difficult; *iii*) models are based on architectures and training practices only up to 2022, and therefore do not fully benefit from recent advances in pre-training techniques. These limitations motivate the development of more robust and up-to-date encoder-based models for Italian.

3. Methodology

In this section, we describe the datasets, tokenizer, model architectures, and training procedure adopted in our experiments.

3.1. Data

The corpora used to train all models in the ALBERTURIN family and GETTONE consist of data collected from a variety of sources selected to build a high-quality general-purpose dataset. The corpus covers multiple domains, ranging from standard resources commonly used for language model pre-training, such as Wikipedia dumps, to social media content and legal documents with a data cut-off for all sources is December 2025. The entire corpus is approximately 380B tokens from which we extracted 100B tokens for this study. Figure 1 shows a visual representation of the data composition for ALBERTURIN and Appendix B provides additional details about all data sources.

All data underwent language validation using fastText.⁹ Due to the dimension of the corpus and

³<https://huggingface.co/dbmdz/bert-base-italian-cased>

⁴<https://huggingface.co/dbmdz/bert-base-italian-xxl-cased>

⁵<https://github.com/idb-ita/GilBERTo>

⁶<https://github.com/musixmatchresearch/umberto>

⁷<https://liberliber.it>

⁸<https://it.wikisource.org>

⁹<https://fasttext.cc/>

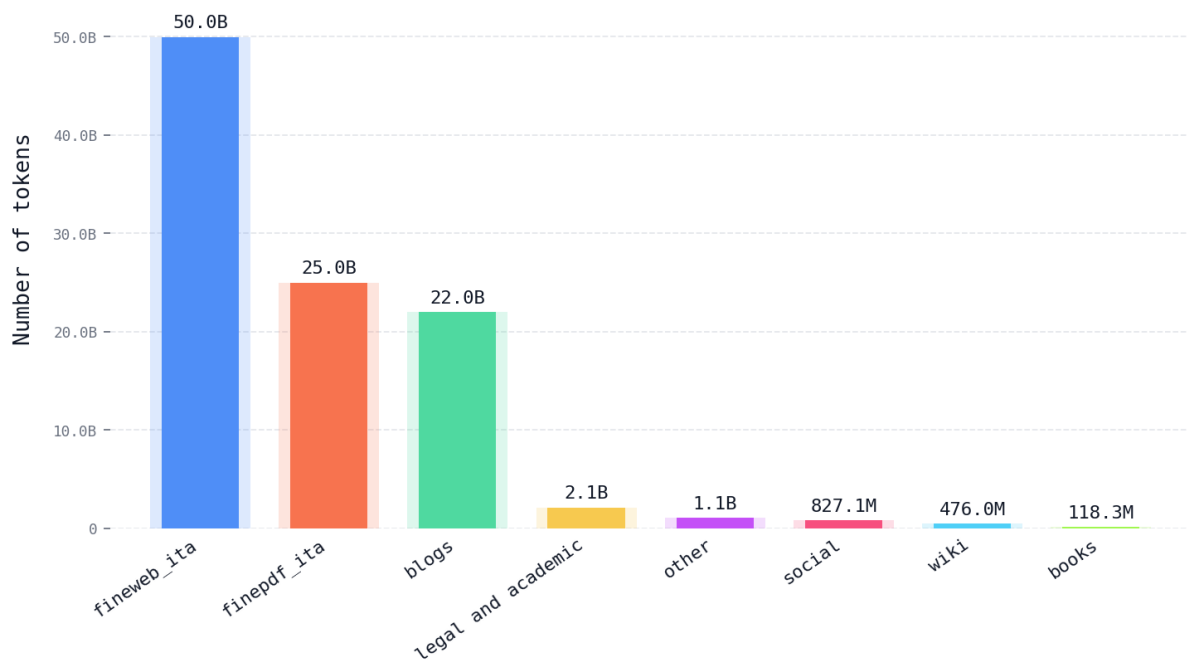


Figure 1: Data composition for ALBERTONE101 divided by source category for a total of approximately 100B tokens.

estimated computing times, we did not apply other more complicated forms of pre-processing such as detoxification. The entire corpus was pre-tokenized to speed up training. Before training, the dataset was shuffled using the same ordering procedure for all model splits. Subsequently, for each model, a portion of the corpus was selected to match a predefined target number of tokens. In particular, the corpus used for the positional embedding experiments contains 3.5B tokens, while the datasets used for ALBERTMINI, ALBERTINA, and ALBERTONE101 contain 7B, 14B, and approximately 100B tokens, respectively.

3.2. Tokenizer

Tokenization is the process of segmenting words into tokens (subwords) that can be processed by a model. A tokenizer is characterized by two main factors: vocabulary size and fertility. The vocabulary size is chosen a priori and determines the number of available subword units. While a larger vocabulary generally reduces token fragmentation, it also increases the number of embedding parameters and thus the overall model size. Fertility, instead, measures the average number of tokens into which a word is split. A lower fertility indicates that the tokenizer represents words more compactly, which in turn reduces sequence length and improves computational efficiency. Ideally, a tokenizer should achieve low fertility without requiring an excessively large vocabulary. Different tokenization algorithms define subword units in different ways. Among the most widely adopted approaches are Byte-Pair Encoding (BPE), which iteratively merges frequent token pairs [49]; WordPiece, which selects merges based on likelihood improvements; and Unigram, which starts from a large vocabulary and iteratively removes subword units according to a probabilistic criterion [50].

We propose GETTONE, a new Italian BPE tokenizer with a vocabulary size of 32,768, and used across all models in the ALBERTURIN family. We compare its fertility against several existing Italian and multilingual cased tokenizers from both encoder- and decoder-only models, including *dbmdz*, EuroBERT, Minerva-350M, Minerva-7B, multilingual BERT (mBERT), UmBERTo, Velvet-2B,¹⁰ and XLM-R.

¹⁰<https://huggingface.co/Almawave/Velvet-2B>

3.3. Architecture and Training

Our goal is to develop a family of Italian BERT models that incorporates recent innovations in Transformer architectures. To this end, we build upon the work of Breton et al. [34] and other recent studies on modern encoder architectures [35, 36]. As pre-training language models is very computationally expensive, we conducted all our training with Matformer,¹¹ a new library developed by us that focuses on memory- and time-efficiency. Starting from the original BERT architecture, we apply several modifications that have been found to improve model performance and efficiency. In particular, we adopt a Pre-LayerNorm architecture and replace LayerNorm with RMSNorm [51], a computationally more efficient normalization strategy with comparable performance. We employ SwiGLU activations [52] and use Muon [53], a recent optimizer designed for efficient large-scale training. We use a mask-only corruption strategy (i.e., without “same token” or “random token” replacements) with a masking rate of 20%. Finally, we incorporate unpadding, a technique that removes unnecessary padding tokens during computation in order to improve training efficiency [54]. We use 12 layers with a hidden size of 768, 12 attention heads, and a feed-forward factor of 3. We additionally employ FlashAttention [55] for more efficient training.

An important aspect in the design of modern encoder architectures concerns the handling of long contexts, which are particularly relevant for applications such as document representation and retrieval. However, extending Transformer models to long sequences significantly increases computational costs. For this reason, we increase the sequence length from the original 512 tokens to 1024 tokens and investigate three alternative positional embedding strategies:

- **Absolute**: the learnable absolute positional embeddings used in the original BERT architecture, where each position is associated with a learned embedding vector;
- **RoPE**: Rotary Position Embeddings [26], which encode positional information through token-wise rotations applied in the attention mechanism, enabling improved relative position modeling;
- **ALiBi**: Attention with Linear Biases [27], which incorporates positional information into the attention scores through distance-dependent linear biases, allowing the model to scale beyond the sequence lengths observed during training.

All three configurations are initially trained on 3.5B tokens in order to evaluate the impact of the different positional embedding strategies. Based on the benchmark results and pseudo-perplexity evaluation, we select the best-performing configuration and use it to train ALBERT_{ITA} on 14B tokens. We choose this scale to satisfy the upper-bound compute-optimal scaling recommendations proposed by Dervishi et al. [56] for encoder models with approximately 140M parameters.

Recent encoder models have also increased in scale, showing that larger encoder architectures can achieve stronger downstream performance. For instance, NeoBERT and ModernBERT have approximately 250M and 395M parameters, respectively. This raises the question of whether similar scaling benefits can also be observed for Italian encoder models. To this end, building upon the architectural choices validated in the previous experiments, we develop ALBERT_{ONE101}, a 450M parameters variant trained on approximately 101B tokens. The model consists of 28 layers with a hidden size of 1024 and 16 attention heads. Moreover, following evidence that higher masking ratios can yield stronger encoder representations, particularly in larger models [57], we increase the masking rate to 28%.

Lastly, we train ALBERT_{MINI}, a smaller 90M-parameter model based on the ALBERT_{ITA} architecture and composed of only 6 layers, using 7B training tokens. All models are trained for one epoch with data parallelism. We include the full details of architectures and training in Table 6 in Appendix C.

We evaluate our models against both Italian-specific and multilingual encoder architectures, including ALBERT_{TO}, BERT-*dbmdz*, BERT-*dbmdz-XXL*, EuroBERT, GiLBERT_{TO}, mBERT, UmBERT_{TO}, and XLM-R.

3.4. Evaluation

To evaluate our models, we adopt both extrinsic and intrinsic evaluation protocols. For the extrinsic evaluation, we consider a set of widely used NLP benchmarks, described in Section 3.4.1. For the

¹¹<https://github.com/mrinaldi97/matformer>

intrinsic evaluation, we employ pseudo-perplexity (Section 3.4.2), which additionally allows us to assess the models’ ability to handle sequences of varying lengths.

3.4.1. NLP Benchmarks

We evaluate our models on various established Italian NLP benchmarks covering classification, sequence labeling, and semantic understanding. The selected tasks span a broad range of linguistic phenomena, including social media pragmatics, lexical semantics, morphosyntax, inference, and acceptability judgments, and rely on datasets released within recent EVALITA¹² campaigns or through curated Italian-language resources. All benchmarks provide official training and testing splits and are annotated according to task-specific guidelines. For tasks that did not explicitly provide distinct training and validation set, we created a random validation split starting from the training set.

For each task, we perform a hyperparameter search over the learning rate by fine-tuning models for three epochs and evaluating performance on the validation set at the end of each epoch. The best performing checkpoint is then selected for final evaluation on the test set. The complete list of fine-tuning hyperparameters is reported in Appendix D. Our primary evaluation metric is the macro-averaged F1 score and final scores are computed by averaging results across five runs. Since our evaluation spans multiple tasks, we additionally report the average score across all tasks as a global indicator of overall model performance. Here, we give a brief overview of each task.

Hate Speech (HS) For hate speech detection we use the dataset released in HaSpeeDe 2 (EVALITA 2020) [58], focusing exclusively on Task A, the binary classification of hate speech (HS vs. not-HS). The benchmark provides 6,839 training tweets and test material from both Twitter (1,263 items) and news headlines (500 items).

Name Entity Recognition (NER) For Named Entity Recognition, we rely on NERMuD [59], the shared task introduced at EVALITA 2023 [60], which focuses on extracting and classifying person, organization, and location entities across multiple domains (news, fiction, political speeches). We use the WikiNews subset of the dataset which is the most balanced for type of entities present.

POS Tagging (POS) For part-of-speech tagging we use PoSTWITA-UD [61], the shared task introduced at EVALITA 2016. The dataset consists of over 6,700 of manually annotated examples. The task is harder compared to other PoS corpora due to the nature of tweets, which often contain slang and non-standard grammatical structures.

Sentiment Analysis Positive Polarity (SA) Sentiment analysis is evaluated using SENTIPOLC 2016 [62], the shared task on sentiment classification in Italian tweets presented at EVALITA 2016 [63]. We focus on subtask 2, where polarity is encoded through two independent binary fields indicating the presence of positive (*opos*) and negative (*oneg*) sentiment. Following [46, 64], we approach the task by performing two separate fine-tuning procedures, one for *opos* and one for *oneg*, and report the average performance across the two subtasks.

Italian Corpus of Linguistic Acceptability (ItaCoLA) Linguistic acceptability is evaluated using ItaCoLA, the Italian Corpus of Linguistic Acceptability [65]. The dataset contains approximately 10,000 Italian sentences extracted from linguistic literature and manually annotated with binary acceptability judgments (acceptable vs. unacceptable), following the same methodology adopted for the English CoLA corpus [66].

¹²<https://www.evalita.it/>

Irony Detection in Italian (IronITA) Irony detection is evaluated using IronITA [67], the shared task organized at EVALITA 2018 [68], which focuses on the automatic classification of irony in Italian tweets. In particular we focus on Task A, which consists of more than 4,700 tweets annotated with the binary irony label.

3.4.2. Pseudo-Perplexity

Perplexity measures how well a language model predicts a text sequence, with lower values indicating better predictive ability [69]. Unlike autoregressive language models, masked language models do not naturally define a left-to-right probability distribution over a sequence. Therefore, the standard perplexity metric cannot be directly computed. *Pseudo-perplexity* (PPL) approximates perplexity by masking each token in the sequence one at a time and measuring the probability assigned to the original token conditioned on the remaining context. Formally, given a sequence $x = (x_1, x_2, \dots, x_T)$, pseudo-perplexity is defined as:

$$\text{PPL}(x) = \exp \left(-\frac{1}{T} \sum_{t=1}^T \log p(x_t \mid x_{\setminus t}) \right),$$

where $x_{\setminus t}$ denotes the sequence with the token x_t masked out, and $p(x_t \mid x_{\setminus t})$ is the probability assigned by the model to the original token. Pseudo-perplexity provides an estimate of how well the model captures the structure and dependencies of natural language. In particular, analyzing pseudo-perplexity across different sequence lengths offers useful insights into the model’s ability to maintain contextual understanding over long inputs, revealing potential degradation in performance as the context window increases.

We extracted sequences of different lengths from two different sources: blogs and books. Both of these datasets have been gathered with the same methodologies used in Section 3.1 but collecting them past the data cut-off of the training set to ensure no data contamination. Since not all positional embedding strategies described in Section 3.3 are able to generalize beyond their training context length, we construct two separate subsets of sequences:

- PPL_{short} , a subset of 1,141 sequences derived from blogs that have a length spanning between 0 and 1,024 tokens.
- PPL_{long} , a subset of 1,472 sequences derived from blogs and books that have a length spanning between 1,024 and 4,096 tokens.

Since pseudo-perplexity depends on the adopted tokenizer, comparisons across models employing different tokenizers are not meaningful. For this reason, we primarily use this metric to compare positional embedding strategies and to analyze how model behavior changes as sequence length increases. All models are evaluated on PPL_{short} , whereas only models employing ALiBi and RoPE are additionally evaluated on PPL_{long} , due to their ability to handle sequences beyond the training context length.

4. Results

In this section, we present our experimental results. We first analyze the compactness of the proposed tokenizer through fertility measurements (Section 4.1), then evaluate the impact of different positional embedding strategies on both downstream benchmarks and pseudo-perplexity (Section 4.2). Finally, we compare ALBERT_{ITA} and ALBERT_{ONE101} against existing Italian and multilingual encoder models across a diverse set of downstream tasks (Section 4.3).

4.1. Tokenizer Fertility

We evaluate the tokenizers introduced in Section 3.2 on the sequences contained in PPL_{short} and PPL_{long} . Table 1 reports the corresponding fertility scores, together with the tokenizer types and vocabulary

Table 1

Tokenizer statistics computed on the PPL_{short} corpus. Fertility measures the average number of subword tokens per word, where lower values indicate more compact tokenization.

Tokenizer	Type	Vocab Size	Fertility ↓
GETTONE	BPE	32,768	1.429
Velvet-2B	BPE	126,976	1.471
UmBERTo	BPE	32,005	1.493
<i>dbmdz</i>	WordPiece	31,102	1.497
Minerva-7B	BPE	51,200	1.560
Minerva-350M	BPE	32,768	1.644
XLM-R	Unigram	250,002	1.663
mBert	WordPiece	119,547	1.719
EuroBERT	BPE	128,000	1.934

Table 2

Downstream benchmark results for the positional embedding experiments.

Model	HS	IRONITA	ITACOLA	NER	POS	SA	AVG
BERT _{absolute}	0.586 ± 0.06	0.578 ± 0.04	0.463 ± 0.00	0.717 ± 0.01	0.865 ± 0.01	0.535 ± 0.10	0.624
BERT _{RoPE}	0.684 ± 0.02	0.692 ± 0.01	0.542 ± 0.01	0.783 ± 0.01	0.893 ± 0.00	0.642 ± 0.05	0.706
BERT _{ALiBi}	0.691 ± 0.03	0.700 ± 0.01	0.548 ± 0.02	0.828 ± 0.01	0.920 ± 0.01	0.627 ± 0.05	0.719

sizes. Among all evaluated tokenizers, GETTONE achieves the lowest fertility score (1.429) despite using a smaller vocabulary than several competing models, such as Velvet-2B, which reaches a fertility of 1.471 with a vocabulary size of 126,976. This suggests that our tokenizer is able to represent Italian words more compactly than existing alternatives while maintaining a moderate vocabulary size. As expected, multilingual tokenizers such as XLM-R, mBERT, and EuroBERT exhibit considerably higher fertility (respectively 1.663, 1.719, and 1.934) despite their very large vocabularies, likely due to the need to share subword units across multiple languages. Overall, these results support the adoption of GETTONE in our models.

4.2. Positional Embedding Experiments

To evaluate the impact of positional encoding strategies, we train three BERT variants differing only in the adopted positional embedding method: learnable absolute embeddings, RoPE, and ALiBi. All models share the same architecture, tokenizer, training setup, and data, allowing us to isolate the contribution of positional encoding.

Table 2 reports the downstream performance of the three positional embedding strategies employed. Both RoPE and ALiBi consistently outperform the learnable absolute positional embeddings across all benchmarks. In particular, ALiBi achieves the best average performance, obtaining the highest scores on five out of six downstream tasks and reaching an overall average F1 score of 0.719, compared to 0.706 for RoPE and 0.624 for absolute positional embeddings. Figure 2 reports pseudo-perplexity as a function of input sequence length on the two test sets, PPL_{short} and PPL_{long} . Absolute positional embeddings exhibit a clear degradation as sequence length increases on PPL_{short} , while they cannot be evaluated on PPL_{long} . Both RoPE and ALiBi display more stable behavior across PPL_{short} ; however, when evaluated on PPL_{long} , RoPE becomes noticeably less stable, exhibiting increasing pseudo-perplexity values, whereas ALiBi shows only a slight increase while maintaining consistent performance across long contexts. Notably, ALiBi achieves the lowest pseudo-perplexity values overall and remains relatively stable even beyond the training sequence length, indicating strong generalization to long-context inputs. These results support the adoption of ALiBi in the final architecture.

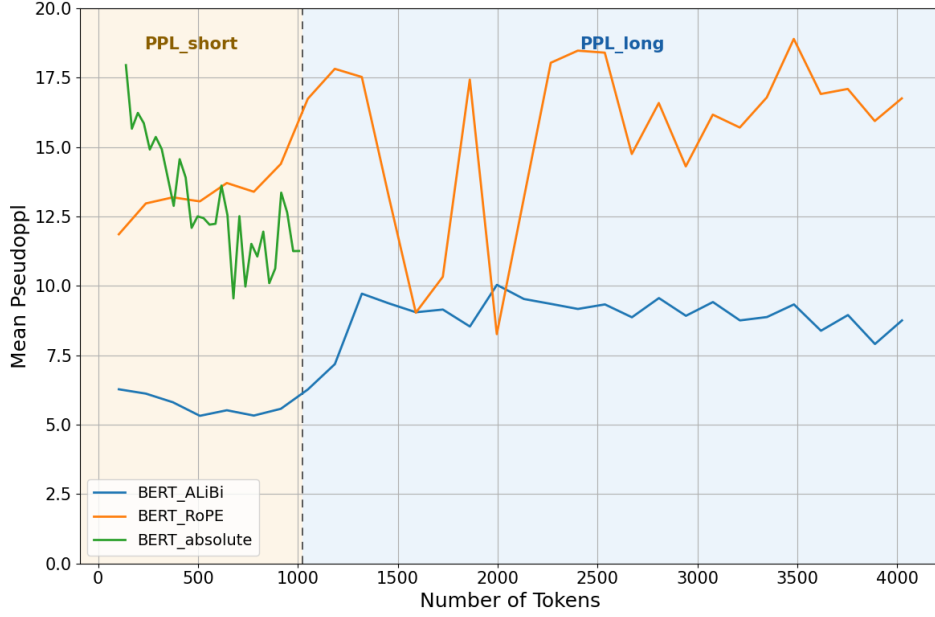


Figure 2: PPL_{short} and PPL_{long} as a function of input sequence length for the experiments conducted on different positional embeddings. Lower values indicate better predictive performance.

Table 3

Downstream benchmark performance of multilingual and Italian encoder models. AVG denotes the average macro-F1 score across all tasks. M, B, and T denote millions, billions, and trillions, respectively. Tokens denotes the total number of tokens seen during training. [†] Estimated from publicly available resources (see Appendix E for details). Models highlighted in green correspond to our proposed models.

Model	Params	Tokens	HS	IRONITA	ITACOLA	NER	POSTWITA	SA	AVG
ALBERTMINI	95M	7B	0.690 ± 0.02	0.707 ± 0.02	0.550 ± 0.05	0.836 ± 0.01	0.917 ± 0.00	0.645 ± 0.04	0.724
GiBERTo	110M	– [†]	0.734 ± 0.01	0.705 ± 0.01	0.577 ± 0.05	0.851 ± 0.01	0.896 ± 0.00	0.655 ± 0.15	0.736
dbmdz	110M	131B [†]	0.752 ± 0.02	0.697 ± 0.01	0.651 ± 0.10	0.881 ± 0.00	0.908 ± 0.00	0.687 ± 0.05	0.763
dbmdz-XXL	110M	786B [†]	0.743 ± 0.01	0.721 ± 0.01	0.725 ± 0.02	0.905 ± 0.00	0.937 ± 0.00	0.716 ± 0.04	0.791
UmBERTo	135M	– [†]	0.739 ± 0.04	0.729 ± 0.01	0.640 ± 0.07	0.531 ± 0.29	0.881 ± 0.02	0.691 ± 0.06	0.702
ALBERTINA	140M	14B	0.672 ± 0.02	0.709 ± 0.00	0.542 ± 0.06	0.863 ± 0.00	0.930 ± 0.00	0.623 ± 0.04	0.723
m-BERT	178M	– [†]	0.724 ± 0.01	0.660 ± 0.03	0.573 ± 0.05	0.889 ± 0.00	0.900 ± 0.00	0.667 ± 0.03	0.735
EuroBERT	210M	5T	0.731 ± 0.02	0.702 ± 0.00	0.291 ± 0.19	0.710 ± 0.16	0.648 ± 0.12	0.619 ± 0.06	0.617
XLM-R	280M	289B [†]	0.700 ± 0.03	0.700 ± 0.01	0.329 ± 0.16	0.884 ± 0.01	0.915 ± 0.00	0.699 ± 0.03	0.705
AIBERTo	282M	16.4B [†]	0.755 ± 0.01	0.736 ± 0.02	0.647 ± 0.01	0.838 ± 0.00	0.901 ± 0.00	0.716 ± 0.02	0.766
ALBERTONE101	450M	101B	0.733 ± 0.01	0.718 ± 0.01	0.689 ± 0.04	0.893 ± 0.00	0.925 ± 0.00	0.693 ± 0.04	0.775

4.3. ALBERTMINI, ALBERTINA, and ALBERTONE101

Table 3 reports the downstream benchmark performance of our models compared to existing Italian and multilingual encoder architectures. Among the evaluated models, BERT-*dbmdz-xxl* achieves the best overall performance with an average of 0.791, highlighting the importance of large-scale training. Among our architectures, ALBERTONE101 obtains the strongest overall performance with an average score of 0.775, while ALBERTMINI and ALBERTINA achieve comparable results with average scores of 0.724 and 0.723, respectively. The performance gap of our models is primarily concentrated on ItaCoLA, while results on the remaining tasks remain broadly competitive. In general, multilingual models such as mBERT, XLM-R, and EuroBERT obtain lower performance than Italian-specific models despite being trained on substantially larger corpora, confirming the advantages of language-specific pre-training. AIBERTo achieves particularly strong results on tweet-based tasks such as Hate Speech, IronITA, and Sentiment Analysis, likely due to its training on a large corpus twitter data. Despite being trained on fewer tokens than several competing architectures, our achieve competitive overall performance.

Figure 3 reports pseudo-perplexity as a function of sequence length for ALBERTMINI, ALBERTINA,

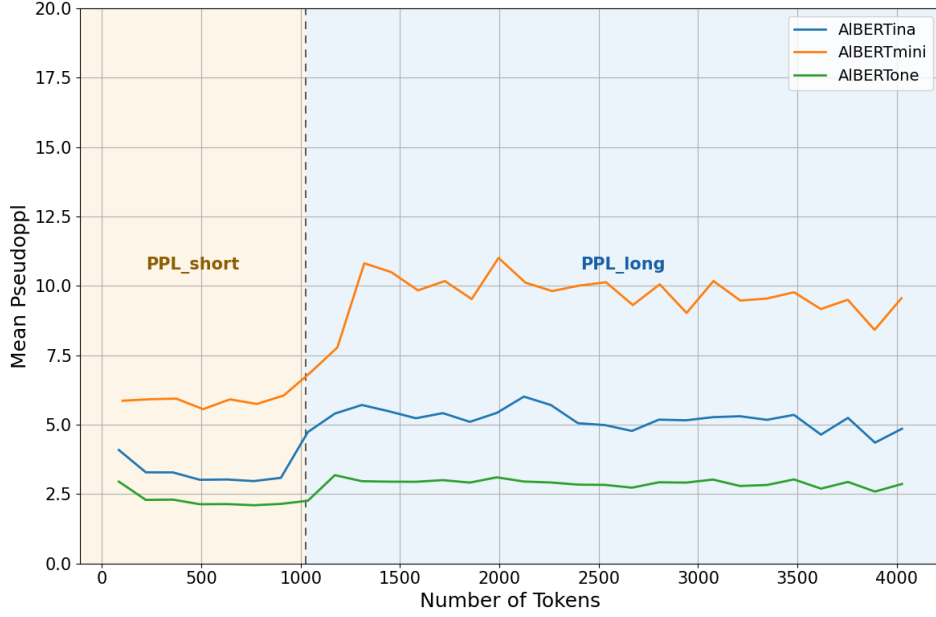


Figure 3: PPL_{short} and PPL_{long} as a function of input sequence length for ALBERTTina and ALBERTone101

and ALBERTone101. As expected, ALBERTone101 consistently achieves the lowest pseudo-perplexity values across both PPL_{short} and PPL_{long} , indicating stronger language modeling capabilities. The difference between ALBERTmini and ALBERTTina becomes particularly evident on PPL_{long} , where the former exhibits a significant increase in pseudo-perplexity as sequence length grows. In contrast, both ALBERTTina and ALBERTone101 maintain more stable behavior across long contexts, suggesting strong generalization beyond the training sequence length.

4.4. Discussion

Overall, our results suggest that recent architectural modifications enable encoder models to achieve competitive downstream performance even when trained on fewer tokens than several existing baselines. To further analyze the practical impact of these architectural choices, we additionally evaluate fine-tuning efficiency. Table 4 highlights the improved training efficiency enabled by recent architectural modifications. Notably, despite having a larger number of parameters, ALBERTTina requires less than half the fine-tuning time of BERT-dbmz models across all downstream tasks, while ALBERTmini requires less than one third of the training time despite having a comparable number of parameters. These results are consistent with the findings of Antoun et al. [70], who report that recent architectural advances can achieve comparable downstream performance while additionally providing improved support for long contexts and faster training and inference. Such efficiency gains are particularly valuable in resource-constrained settings, where reducing training time lowers experimentation costs and hardware requirements.

At the same time, our results suggest that the models have not yet fully saturated. Training loss trends indicate that additional performance gains may still be achievable. However, in this work we intentionally employ training scales aligned with the upper compute-optimal regime proposed by Dervishi et al. [56] for ALBERTmini and ALBERTTina, while ALBERTone101 was trained on approximately 100B tokens, which was the largest scale feasible given our computational resources. Although future improvements are likely to be more incremental, we plan to continue scaling training in future work and to explore similar architectural choices for decoder-only models.

An observation emerging from the benchmark results is that current evaluation settings may not fully capture differences between encoder models, particularly with respect to long-context capabilities. This is evident from the comparison between ALBERTmini and ALBERTTina, which achieve similar

Table 4

Fine-tuning runtimes for each downstream task and model on a single NVIDIA A40 GPU, reported in hours, minutes, and seconds.

Model	Params	HS	IRONITA	ITACOLA	NER	POSTWITA	SA	TOTAL
ALBERT _{MINI}	95M	01m 57s	01m 11s	04m 32s	04m 03s	01m 57s	04m 14s	17m 56s
GilBERT _o	110M	07m 33s	04m 33s	13m 09s	15m 01s	07m 16s	16m 22s	1h 03m 57s
<i>dbmdz</i>	110M	07m 50s	04m 42s	13m 27s	15m 23s	07m 29s	16m 44s	1h 05m 39s
<i>dbmdz-XXL</i>	110M	07m 46s	04m 43s	13m 23s	15m 22s	07m 29s	16m 54s	1h 05m 39s
UmBERT _o	135M	07m 32s	04m 32s	13m 13s	15m 00s	07m 18s	16m 22s	1h 03m 59s
ALBERT _{TINA}	140M	03m 17s	02m 00s	07m 10s	06m 37s	03m 13s	07m 08s	29m 27s
m-BERT	178M	08m 30s	05m 14s	15m 09s	16m 26s	08m 07s	18m 09s	1h 11m 38s
EuroBERT	210M	11m 33s	07m 01s	20m 00s	22m 19s	10m 55s	24m 54s	1h 36m 45s
XLM-R	280M	09m 02s	05m 47s	17m 24s	17m 11s	08m 46s	19m 41s	1h 17m 52s
ALBERT _o	282M	08m 30s	05m 20s	15m 13s	16m 16s	08m 04s	18m 12s	1h 11m 39s
ALBERT _{ONE101}	450M	10m 58s	06m 56s	19m 28s	20m 52s	10m 30s	23m 44s	1h 32m 29s

downstream benchmark results despite exhibiting markedly different pseudo-perplexity trends, especially on long-context evaluations. More generally, downstream results across monolingual Italian models remain relatively close, suggesting that current benchmarks may provide limited differentiation and may also be strongly influenced by domain specialization, as illustrated by the strong performance of ALBERT_o on tweet-based tasks. Moreover, current Italian NLP benchmarks mostly rely on short-text classification tasks and therefore may not fully reflect the advantages of long-context architectures. The benefits of ALiBi and extended context windows are likely to emerge more clearly in retrieval, document-level understanding, and embedding-based applications.

5. Conclusions

In this paper, we presented a new family of open encoder models for the Italian language, ALBERT_{TURIN}, that integrate the most recent advances in encoder architectures. Starting from the original BERT architecture, we integrated several modern improvements, including Pre-LayerNorm with RMSNorm, a mask-only corruption strategy with higher masking rates, and the Muon optimizer for more efficient large-scale training. We also introduced a new BPE tokenizer, GETTONE. Compared to existing Italian tokenizers, it achieves lower fertility while maintaining a moderate vocabulary size. For training, we used a new Italian large-scale corpus composed of texts from multiple domains. Unlike many existing Italian datasets, it reduces reliance on unfiltered web data by prioritizing more curated sources, including blogs, legal documents, and Wikipedia. This diversity of domains aims to support the development of models that generalize well across a wide range of downstream tasks.

One of our main goals was to design models capable of supporting long-context sequences. To this end, we initially evaluated three positional encoding strategies: absolute positional embeddings, RoPE, and ALiBi. Our experiments showed that ALiBi consistently improved both downstream task performance and pseudo-perplexity. Building on these findings, we trained three models: ALBERT_{MINI}, ALBERT_{TINA} and ALBERT_{ONE101}. ALBERT_{MINI} is a 95M-parameter model trained on 7B tokens, ALBERT_{TINA} is a 140M-parameter model trained on 14B tokens, and ALBERT_{ONE101} is a 450M-parameter model trained on approximately 100B tokens. While our models support substantially longer input sequences, current Italian encoder benchmark suites do not include standardized long-context downstream tasks. For this reason, we evaluate on the established benchmark suite adopted by prior Italian encoder models to ensure a fair comparison: the proposed models achieve competitive performance considering their training scale. Furthermore, their support for long-context processing makes them particularly suitable for tasks involving long input sequences, such as long-document classification and retrieval. At the same time, the architectural choices adopted improve training efficiency, enabling faster experimentation and lowering the computational requirements needed to develop and evaluate encoder models. Finally, we note that existing Italian encoder models lack comprehensive documentation regarding crucial aspects such as training data, number of training tokens, or the code used for training and evaluation.

In contrast, we place strong emphasis on transparency and reproducibility by documenting the key components, design choices, and resources that enabled this work.

Some directions remain open for future work. First, current Italian encoder benchmarks are largely centered on short-text classification tasks and may therefore underestimate the advantages of long-context encoder architectures. In future work, we plan to evaluate our models on more challenging long-document benchmarks, drawing inspiration from recent evaluation settings proposed for decoder-only models [71, 72]. Second, although our models are partly motivated by embedding and retrieval applications, we do not yet evaluate them on dedicated semantic similarity or embedding benchmarks. To this end, future work could investigate contrastive learning approaches, leveraging existing resources such as those introduced by Novello and Tamburini [73] while also developing new resources. A limitation of our work is that, due to computational constraints, we do not provide a complete ablation study isolating the contribution of each architectural modification. A more systematic investigation of the impact of these design choices is left for future work.

Acknowledgments

We thank HPC4AI [74], the High-Performance Computing for Artificial Intelligence center at the University of Turin, for providing the computational resources required for this work.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT 5.5 and Claude Sonnet 4.6 in order to: Grammar and spelling check and Rephrasing. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication’s content.

References

- [1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, *Advances in neural information processing systems* 30 (2017).
- [2] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, *arXiv preprint arXiv:1409.0473* (2014).
- [3] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al., Language models are unsupervised multitask learners, *OpenAI blog* 1 (2019) 9.
- [4] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, *Advances in neural information processing systems* 33 (2020) 1877–1901.
- [5] A. Grattaflori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, et al., The llama 3 herd of models, *arXiv preprint arXiv:2407.21783* (2024).
- [6] D. Guo, D. Yang, H. Zhang, J. Song, P. Wang, Q. Zhu, R. Xu, R. Zhang, S. Ma, X. Bi, et al., Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, *arXiv preprint arXiv:2501.12948* (2025).
- [7] K. Team, Y. Bai, Y. Bao, Y. Charles, C. Chen, G. Chen, H. Chen, H. Chen, J. Chen, N. Chen, et al., Kimi k2: Open agentic intelligence, *arXiv preprint arXiv:2507.20534* (2025).
- [8] P. Rajpurkar, R. Jia, P. Liang, Know what you don’t know: Unanswerable questions for squad, in: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2018, pp. 784–789.
- [9] Y. Wang, X. Ma, G. Zhang, Y. Ni, A. Chandra, S. Guo, W. Ren, A. Arulraj, X. He, Z. Jiang, et al., Mmlu-pro: A more robust and challenging multi-task language understanding benchmark, *Advances in Neural Information Processing Systems* 37 (2024) 95266–95290.

- [10] C. E. Jimenez, J. Yang, A. Wettig, S. Yao, K. Pei, O. Press, K. R. Narasimhan, SWE-bench: Can language models resolve real-world github issues?, in: The Twelfth International Conference on Learning Representations, 2024. URL: <https://openreview.net/forum?id=VTF8yNQm66>.
- [11] T. Y. Zhuo, V. M. Chien, J. Chim, H. Hu, W. Yu, R. Widyasari, I. N. B. Yusuf, H. Zhan, J. He, I. Paul, S. Brunner, C. GONG, J. Hoang, A. R. Zebaze, X. Hong, W.-D. Li, J. Kaddour, M. Xu, Z. Zhang, P. Yadav, N. Jain, A. Gu, Z. Cheng, J. Liu, Q. Liu, Z. Wang, D. Lo, B. Hui, N. Muennighoff, D. Fried, X. Du, H. de Vries, L. V. Werra, Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions, in: The Thirteenth International Conference on Learning Representations, 2025. URL: <https://openreview.net/forum?id=YrycTjllL0>.
- [12] D. Rein, B. L. Hou, A. C. Stickland, J. Petty, R. Y. Pang, J. Dirani, J. Michael, S. R. Bowman, Gpqa: A graduate-level google-proof q&a benchmark, arXiv preprint arXiv:2311.12022 (2023).
- [13] E. Glazer, E. Erdil, T. Besiroglu, D. Chicharro, E. Chen, A. Gunning, C. F. Olsson, J.-S. Denain, A. Ho, E. d. O. Santos, et al., Frontiermath: A benchmark for evaluating advanced mathematical reasoning in ai, arXiv preprint arXiv:2411.04872 (2024).
- [14] V. Barres, H. Dong, S. Ray, X. Si, K. Narasimhan, τ^2 -Bench: Evaluating Conversational Agents in a Dual-Control Environment, arXiv preprint arXiv:2506.07982 (2025).
- [15] T. Patwardhan, R. Dias, E. Proehl, G. Kim, M. Wang, O. Watkins, S. P. Fishman, M. Aljubei, P. Thacker, L. Fauconnet, et al., Gdpval: Evaluating ai model performance on real-world economically valuable tasks, arXiv preprint arXiv:2510.04374 (2025).
- [16] J. Devlin, M.-W. Chang, K. Lee, K. 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), 2019, pp. 4171–4186.
- [17] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, H. Wang, et al., Retrieval-augmented generation for large language models: A survey, arXiv preprint arXiv:2312.10997 2 (2023) 32.
- [18] K. Shuster, S. Poff, M. Chen, D. Kiela, J. Weston, Retrieval augmentation reduces hallucination in conversation, in: Findings of the Association for Computational Linguistics: EMNLP 2021, 2021, pp. 3784–3803.
- [19] S. Nie, F. Zhu, Z. You, X. Zhang, J. Ou, J. Hu, J. Zhou, Y. Lin, J.-R. Wen, C. Li, Large language diffusion models, Advances in Neural Information Processing Systems 38 (2026) 50608–50646.
- [20] L. Martin, B. Muller, P. O. Suarez, Y. Dupont, L. Romary, É. V. de La Clergerie, D. Seddah, B. Sagot, Camembert: a tasty french language model, in: Proceedings of the 58th annual meeting of the association for computational linguistics, 2020, pp. 7203–7219.
- [21] M. Rinaldi, R. Varvara, V. Patti, et al., Testimole-conversational: A 30-billion-word italian discussion board corpus (1996–2024) for language modeling and sociolinguistic research, in: Proceedings of the 12th Workshop on Challenges in the Management of Large Corpora (CMLC-12)@ LREC 2026, ELRA Language Resources Association, 2026, pp. 1–11.
- [22] S. Balloccu, P. Schmidtová, M. Lango, O. Dušek, Leak, cheat, repeat: Data contamination and evaluation malpractices in closed-source llms, in: Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), 2024, pp. 67–93.
- [23] T. Olmo, :, A. Ettinger, A. Bertsch, B. Kuehl, D. Graham, D. Heineman, D. Groeneveld, F. Brahman, F. Timbers, H. Ivison, J. Morrison, J. Poznanski, K. Lo, L. Soldaini, M. Jordan, M. Chen, M. Noukhovitch, N. Lambert, P. Walsh, P. Dasigi, R. Berry, S. Malik, S. Shah, S. Geng, S. Arora, S. Gupta, T. Anderson, T. Xiao, T. Murray, T. Romero, V. Graf, A. Asai, A. Bhagia, A. Wettig, A. Liu, A. Rangapur, C. Anastasiades, C. Huang, D. Schwenk, H. Trivedi, I. Magnusson, J. Lochner, J. Liu, L. J. V. Miranda, M. Sap, M. Morgan, M. Schmitz, M. Guerquin, M. Wilson, R. Huff, R. L. Bras, R. Xin, R. Shao, S. Skjonsberg, S. Z. Shen, S. S. Li, T. Wilde, V. Pyatkin, W. Merrill, Y. Chang, Y. Gu, Z. Zeng, A. Sabharwal, L. Zettlemoyer, P. W. Koh, A. Farhadi, N. A. Smith, H. Hajishirzi, Olmo 3, 2026. URL: <https://arxiv.org/abs/2512.13961>. arXiv:2512.13961.
- [24] P. Apertus, A. Hernández-Cano, A. Hägele, A. H. Huang, A. Romanou, A.-J. Solergibert, B. Pasztor,

- B. Messmer, D. Garbaya, E. F. Āurech, et al., Apertus: Democratizing open and compliant llms for global language environments, arXiv preprint arXiv:2509.14233 (2025).
- [25] R. Orlando, L. Moroni, P.-L. H. Cabot, S. Conia, E. Barba, S. Orlandini, G. Fiameni, R. Navigli, Minerva llms: The first family of large language models trained from scratch on italian data, in: Proceedings of the 10th Italian conference on computational linguistics (CLiC-it 2024), 2024, pp. 707–719.
 - [26] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, Y. Liu, Roformer: Enhanced transformer with rotary position embedding, *Neurocomputing* 568 (2024) 127063.
 - [27] O. Press, N. A. Smith, M. Lewis, Train short, test long: Attention with linear biases enables input length extrapolation, arXiv preprint arXiv:2108.12409 (2021).
 - [28] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781 (2013).
 - [29] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, in: M. Walker, H. Ji, A. Stent (Eds.), Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 2227–2237. URL: <https://aclanthology.org/N18-1202/>. doi:10.18653/v1/N18-1202.
 - [30] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized bert pretraining approach, arXiv preprint arXiv:1907.11692 (2019).
 - [31] D. Samuel, A. Kutuzov, L. Øvrelid, E. Velldal, Trained on 100 million words and still in shape: Bert meets british national corpus, in: Findings of the Association for Computational Linguistics: EACL 2023, 2023, pp. 1954–1974.
 - [32] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, V. Stoyanov, Unsupervised cross-lingual representation learning at scale, in: Proceedings of the 58th annual meeting of the association for computational linguistics, 2020, pp. 8440–8451.
 - [33] Z. Nussbaum, J. X. Morris, A. Mulyar, B. Duderstadt, Nomic embed: Training a reproducible long context text embedder, *Transactions on Machine Learning Research* (2025). Reproducibility Certification.
 - [34] L. L. Breton, Q. Fournier, J. X. Morris, M. E. Mezouar, S. Chandar, NeoBERT: A next generation BERT, *Transactions on Machine Learning Research* (2025). Reproducibility Certification.
 - [35] B. Warner, A. Chaffin, B. Clavié, O. Weller, O. Hallström, S. Taghadouini, A. Gallagher, R. Biswas, F. Ladhak, T. Aarsen, et al., Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference, in: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2025, pp. 2526–2547.
 - [36] N. Boizard, H. Gisserot-Boukhlef, D. M. Alves, A. Martins, A. Hammal, C. Corro, C. HUDELLOT, E. Malherbe, E. Malaboeuf, F. Jourdan, G. Hautreux, J. Alves, K. E. Haddad, M. Faysse, M. Peyrard, N. M. Guerreiro, P. Fernandes, R. Rei, P. Colombo, EuroBERT: Scaling multilingual encoders for european languages, in: Second Conference on Language Modeling, 2025.
 - [37] M. Ulčar, A. Žagar, C. S. Armendariz, A. Repar, S. Pollak, M. Purver, M. Robnik-Šikonja, Evaluation of contextual embeddings on less-resourced languages, arXiv preprint arXiv:2107.10614 (2021).
 - [38] L. G. A. Zeldes, Microbert: Effective training of low-resource monolingual bert models through parameter reduction and multitask learning, *MRL* 2022 (2022) 86.
 - [39] V. Basile, L. Bioglio, A. Bosca, C. Bosco, V. Patti, Uinaul: A unified benchmark for italian natural language understanding, in: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), 2023, pp. 348–356.
 - [40] F. Tamburini, How “bertology” changed the state-of-the-art also for italian nlp, in: Proceedings of the Seventh Italian Conference on Computational Linguistics (CLiC-it 2020), 2020, pp. 313–319.
 - [41] P. J. O. Suárez, B. Sagot, L. Romary, Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures, in: 7th Workshop on the Challenges in the Management of Large

Corpora (CMLC-7), Leibniz-Institut für Deutsche Sprache, 2019.

- [42] M. Muffo, E. Bertino, Bertino: An italian distilbert model, in: Proceedings of the Seventh Italian Conference on Computational Linguistics (CLiC-it 2020), 2020, pp. 429–434.
- [43] V. Sanh, L. Debut, J. Chaumond, T. Wolf, Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, arXiv preprint arXiv:1910.01108 (2019).
- [44] V. Lyding, E. Stemle, C. Borghetti, M. Brunello, S. Castagnoli, F. Dell’Orletta, H. Dittmann, A. Lenci, V. Pirrelli, et al., The paisa’corpus of italian web texts, in: Proceedings of the 9th web as corpus workshop (WaC-9)@ EACL 2014, EACL (European chapter of the Association for Computational Linguistics), 2014, pp. 36–43.
- [45] M. Baroni, S. Bernardini, A. Ferraresi, E. Zanchetta, The wacky wide web: a collection of very large linguistically processed web-crawled corpora, Language resources and evaluation 43 (2009) 209–226.
- [46] M. Polignano, P. Basile, M. De Gemmis, G. Semeraro, V. Basile, et al., Alberto: Italian bert language understanding model for nlp challenging tasks based on tweets, in: CEUR workshop proceedings, volume 2481, CEUR, 2019, pp. 1–6.
- [47] V. Basile, M. Lai, M. Sanguinetti, et al., Long-term social media data collection at the university of turin, in: Proceedings of the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), CEUR-WS, 2018, pp. 1–6.
- [48] A. P. Aprosio, S. Menini, S. Tonelli, Bertoldo, the historical bert for italian, in: Proceedings of the second workshop on language technologies for historical and ancient languages, 2022, pp. 68–72.
- [49] R. Sennrich, B. Haddow, A. 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), 2016, pp. 1715–1725.
- [50] T. 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), 2018, pp. 66–75.
- [51] B. Zhang, R. Sennrich, Root mean square layer normalization, Advances in neural information processing systems 32 (2019).
- [52] N. Shazeer, Glu variants improve transformer, arXiv preprint arXiv:2002.05202 (2020).
- [53] J. Liu, J. Su, X. Yao, Z. Jiang, G. Lai, Y. Du, Y. Qin, W. Xu, E. Lu, J. Yan, et al., Muon is scalable for llm training, arXiv preprint arXiv:2502.16982 (2025).
- [54] J. Zeng, M. Li, Z. Wu, J. Liu, Y. Liu, D. Yu, Y. Ma, Boosting distributed training performance of the unpadded bert model, CoRR (2022).
- [55] T. Dao, Flashattention-2: Faster attention with better parallelism and work partitioning, in: International Conference on Learning Representations, volume 2024, 2024, pp. 35549–35562.
- [56] M. Dervishi, A. Allauzen, G. Synnaeve, Y. Lecun, Training compute-optimal transformer encoder models, in: Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, 2025, pp. 35602–35617.
- [57] A. Wettig, T. Gao, Z. Zhong, D. Chen, Should you mask 15% in masked language modeling?, in: Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, 2023, pp. 2985–3000.
- [58] M. Sanguinetti, G. Comandini, E. Di Nuovo, S. Frenda, M. Stranisci, C. Bosco, T. Caselli, V. Patti, I. Russo, Haspeede 2@ evalita2020: Overview of the evalita 2020 hate speech detection task, Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (2020).
- [59] A. Palmero Aprosio, T. Paccosi, et al., Nermud at evalita 2023: overview of the named-entities recognition on multi-domain documents task, in: CEUR WORKSHOP PROCEEDINGS, volume 3473, CEUR-WS, 2023.
- [60] M. Lai, S. Menini, M. Polignano, V. Russo, R. Sprugnoli, Evalita 2023: Overview of the 8th evaluation campaign of natural language processing and speech tools for italian, in: EVALITA Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian Final Workshop: Parma, Italy, September 7-8th, 2023, Accademia University Press, 2024, p. 3.
- [61] C. Bosco, F. Tamburini, A. Bolioli, A. Mazzei, et al., Overview of the evalita 2016 part of speech

- on twitter for italian task, in: CEUR Workshop Proceedings, volume 1749, CEUR Workshop Proceedings (CEUR-WS. org), 2016, pp. 1–7.
- [62] F. Barbieri, V. Basile, D. Croce, M. Nissim, N. Novielli, V. Patti, et al., Overview of the evalita 2016 sentiment polarity classification task, in: CEUR workshop proceedings, volume 1749, CEUR-WS, 2016.
 - [63] P. Basile, F. Cutugno, M. Nissim, V. Patti, R. Sprugnoli, et al., Evalita 2016: Overview of the 5th evaluation campaign of natural language processing and speech tools for italian, in: CEUR Workshop Proceedings, volume 1749, CEUR-WS, 2016, pp. 1–4.
 - [64] F. Tamburini, How “BERTology” changed the state-of-the-art also for Italian NLP, in: J. Monti, F. Dell’Orletta, F. Tamburini (Eds.), Proceedings of the Seventh Italian Conference on Computational Linguistics (CLiC-it 2020), CEUR Workshop Proceedings, Bologna, Italy, 2020, pp. 313–319. URL: <https://aclanthology.org/2020.clicit-1.48/>.
 - [65] D. Trotta, R. Guarasci, E. Leonardelli, S. Tonelli, Monolingual and cross-lingual acceptability judgments with the italian cola corpus, arXiv preprint arXiv:2109.12053 (2021).
 - [66] A. Warstadt, A. Singh, S. R. Bowman, Neural network acceptability judgments, Transactions of the Association for Computational Linguistics 7 (2019) 625–641.
 - [67] A. T. Cignarella, S. Frenda, V. Basile, C. Bosco, V. Patti, P. Rosso, et al., Overview of the evalita 2018 task on irony detection in italian tweets (ironita), in: CEUR Workshop Proceedings, volume 2263, CEUR-WS, 2018, pp. 1–6.
 - [68] T. Caselli, V. Patti, N. Novielli, U. d. S. di Bari Aldo, P. Rosso, Evalita 2018: Overview on the 6th evaluation campaign of natural language processing and speech tools for italian, in: Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018), 2018, pp. 3–8.
 - [69] F. Jelinek, R. L. Mercer, L. R. Bahl, J. K. Baker, Perplexity—a measure of the difficulty of speech recognition tasks, The journal of the Acoustical Society of America 62 (1977) S63–S63.
 - [70] W. Antoun, B. Sagot, D. Seddah, Modernbert or debertav3? examining architecture and data influence on transformer encoder models performance, in: Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, 2025, pp. 3061–3074.
 - [71] B. Magnini, R. Zanolli, M. Resta, M. Cimmino, P. Albano, M. Madeddu, V. Patti, Evalita-llm: Benchmarking large language models on italian, arXiv preprint arXiv:2502.02289 (2025).
 - [72] B. Magnini, M. Madeddu, M. Resta, R. Zanolli, M. Cimmino, P. Albano, V. Patti, A leaderboard for benchmarking llms on italian, in: Proceedings of the Eleventh Italian Conference on Computational Linguistics (CLiC-it 2025), 2025, pp. 636–646.
 - [73] Y. Noviello, F. Tamburini, Exploring text-embedding retrieval models for the italian language, in: Proceedings of the 10th Italian Conference on Computational Linguistics (CLiC-it 2024), 2024, pp. 654–661.
 - [74] M. Aldinucci, S. Rabellino, M. Pironti, F. Spiga, P. Viviani, M. Drocco, M. Guerzoni, G. Boella, M. Mellia, P. Margara, I. Drago, R. Marturano, G. Marchetto, E. Piccolo, S. Bagnasco, S. Lusso, S. Vallero, G. Attardi, A. Barchiesi, A. Colla, F. Galeazzi, HPC4AI: an ai-on-demand federated platform endeavour, in: D. R. Kaeli, M. Pericàs (Eds.), Proceedings of the 15th ACM International Conference on Computing Frontiers, CF 2018, Ischia, Italy, May 08-10, 2018, ACM, 2018, pp. 279–286. URL: <https://doi.org/10.1145/3203217.3205340>. doi:10.1145/3203217.3205340.
 - [75] G. Penedo, H. Kydliček, A. Lozhkov, M. Mitchell, C. Raffel, L. Von Werra, T. Wolf, et al., The fineweb datasets: Decanting the web for the finest text data at scale, Advances in Neural Information Processing Systems 37 (2024) 30811–30849.
 - [76] H. Kydliček, G. Penedo, L. von Werra, Finepdfs, <https://huggingface.co/datasets/HuggingFaceFW/finepdfs>, 2025.

A. Overview of Existing Encoder Models

Table 5 provides a concise overview of the encoder models discussed in the related work. For each model, we report the supported language, the maximum context length, and its main distinguishing characteristic.

Table 5

Overview of the encoder models discussed in Section 2.

Model	Language	Context Length	Notes
BERT	EN	512	Original encoder architecture
RoBERTa	EN	512	Removes NSP and extends training
XLM-R	Multi	512	Multilingual RoBERTa
NomicBERT	EN	2048	Long-context encoder
NeoBERT	EN	4096	Modernized BERT architecture
ModernBERT	EN	8192	Large-scale modern encoder
EuroBERT	Multi	8192	Modern multilingual encoder
BERT-dbmz	IT	512	Standard Italian BERT
GilBERTo	IT	512	Italian RoBERTa
UmBERTo	IT	512	Italian RoBERTa with WWM
BERTino	IT	512	Italian DistilBERT model
AlBERTo	IT	128	Italian Twitter-specific encoder
BERToldo	IT	512	Historical Italian encoder

B. Dataset Details

The dataset used for pre-training the ALBERTURIN models contains the following sources:

- *Fineweb* [75], which is a refined collections of webtext which we filtered for Italian.
- *FinePDF* [76], from which we used the Italian portion.
- *Blogs and Forums*, comprising of scraped data from forums and usenet sites [21]. In addition, we manually selected a list of high quality blogs that touch different areas including politics, news, science and more.
- *Wiki*, which includes a dump of Wikipedia in the Italian language and, a collection of content under the Wikimedia umbrella regarding variations of the Italian language (e.g., dialects) and Latin.
- *Social*, consisting mainly of Twita [47], a large-scale collection of Twitter posts ranging from 2014 until the API shutdown in 2023.
- *Legal and Academic*, several sources in more formal domains. For the legal domain, we use *Gazzetta Ufficiale*,¹³ the archive of the Italian legislative acts and *Eurlex*¹⁴ for European laws. From the political domain we use *ParlaMint*,¹⁵ a corpus of parliamentary debates of different countries in Europe. academic. Meanwhile, we gathered open access academic texts mainly from *Open Journal Systems* and *Iris* of different Italian universities.
- *Books* from *LiberLiber*,¹⁶ a collection of copyright-free books in Italian.
- *Other data*, mainly made of smaller corpora that regard Italian texts from transcriptions.

¹³<https://www.gazzettaufficiale.it/>

¹⁴<https://eur-lex.europa.eu>

¹⁵<https://www.clarin.eu/parlamint>

¹⁶<https://liberliber.it/>

C. Architectural Details

Table 6 summarizes the architectural and training characteristics of ALBERT_{TINA} and ALBERT_{TONE101}. ALBERT_{MINI} follows the same architecture as ALBERT_{TINA}, differing only in the number of layers, using 6 instead of 12.

ALBERT_{TINA} Training is performed for one epoch using the Muon optimizer with a custom learning rate scheduler. The peak learning rate is set to 5×10^{-4} , while the final learning rate is 2×10^{-5} . We employ a warmup phase corresponding to 5% of the total training steps, without any hold phase. Weight decay is set to 0.01, and gradients are clipped to 1.0.

Table 6

Architectural and training details of ALBERT_{MINI}, ALBERT_{TINA}, and ALBERT_{TONE101}.

	ALBERT _{MINI}	ALBERT _{TINA}	ALBERT _{TONE101}
Layers	6	12	28
Hidden Size	768	768	1024
Depth-to-Width Ratio ($\times 100$)	0.78	1.56	2.73
Attention Heads	12	12	16
Parameters	95M	140M	450M
Activation Function	SwiGLU	SwiGLU	SwiGLU
Positional Encoding	ALiBi	ALiBi	ALiBi
Normalization	Pre-RMSNorm	Pre-RMSNorm	Pre-RMSNorm
Vocabulary Size	32K	32K	32K
Sequence Length	1,024	1,024	1,024
Masking Rate	20%	20%	28%
Masking Scheme	100% mask	100% mask	100% mask
Tie Word Embeddings	False	False	True
Optimizer	Muon	Muon	Muon
Scheduler	Custom	Custom	Custom
Warmup	5% of total steps	5% of total steps	7% of total steps
Peak Learning Rate	5×10^{-4}	5×10^{-4}	5×10^{-4}
Final Learning Rate	2×10^{-5}	2×10^{-5}	1×10^{-5}
Batch Size	48	48	32
Gradient Accumulation	16	16	64
Gradient Clipping	1.0	1.0	1.0
Nodes - GPUs	1 - 4 NVIDIA A40	1 - 4 NVIDIA A40	4 - 8 NVIDIA A100
Epochs	1	1	1
Total Training Tokens	7B	14B	101B
Training Time	4.10h	17.45h	108.30h

D. Fine-Tuning Hyperparameters

Models were trained for 3 epochs using the AdamW optimizer, a sequence length of 128, and a batch size of 64. Learning rates were selected through grid search over $\{10^{-4}, 10^{-5}, 2 \times 10^{-5}, 5 \times 10^{-5}\}$. Validation performance was monitored at each epoch, and the best-performing checkpoint was used for testing. Fine-tuning was performed on an NVIDIA A40 GPU.

E. Training Token Estimates

For some baseline models, the total number of training tokens is not explicitly reported by the authors. In these cases, we estimate the number of processed tokens from the publicly available resources.

ALBERTo Polignano et al. [46] report a training batch size of 128, a maximum sequence length of 128, and a total of 1,000,000 training steps. Following standard practice, we estimate the total number

of processed tokens as:

$$1,000,000 \times 128 \times 128 = 16,384,000,000,$$

corresponding to approximately 16.4 billion training tokens.

BERT-dbmdz and dbmdz-XXL The Hugging Face model page¹⁷ reports training with a sequence length of 512 for approximately 2–3 million steps on Italian Wikipedia and OPUS data. Since the batch size is not explicitly reported for the BERT models, we assume a batch size of 128, following the configuration reported for the associated ELECTRA models and common practice at the time. Under this assumption, we estimate that the standard models processed approximately 131–197 billion training tokens, computed as:

$$512 \times 128 \times (2-3)\text{M}.$$

For the XXL model, the same source reports an extended corpus including the Italian portion of OSCAR, increasing the corpus size from approximately 2.05 billion to 13.14 billion tokens. Assuming a training schedule proportional to the corpus scaling factor (approximately $6\times$ larger than the standard setup), we estimate a total of 12–18 million training steps. Under the same assumptions on sequence length and batch size, this corresponds to approximately 786 billion to 1.18 trillion processed training tokens, computed as:

$$512 \times 128 \times (12-18)\text{M}.$$

EuroBERT Boizard et al. [36] report training EuroBERT on a multilingual corpus containing approximately 5 trillion tokens, including 4.8 trillion tokens used for pre-training and 200 billion tokens used during the annealing phase.

GilBERTo GilBERTo¹⁸ is trained on approximately 71GB of Italian text extracted from the OSCAR corpus, corresponding to about 11.25 billion words. The model was trained for 100k steps using a vocabulary of 32k BPE subwords generated with SentencePiece. Since the training batch size is not explicitly reported, we do not estimate the total number of processed training tokens.

mBERT mBERT [16] was trained on the Wikipedia corpora of the top 104 languages using a shared multilingual vocabulary. Since the authors do not report the total number of training tokens or training steps, we do not provide a token-count estimate.

UmBERTo UmBERTo¹⁹ was trained on the Italian portion of OSCAR/CommonCrawl, corresponding to approximately 69GB of textual data. Since the authors do not report the total number of training steps or tokens, we do not provide a token-count estimate.

XLM-R XLM-R [32] was trained on the CC-100 corpus, a multilingual dataset covering 100 languages extracted from CommonCrawl. Based on the corpus statistics reported by the authors in the appendix of the paper, the dataset contains approximately 288,703 million tokens, corresponding to about 289 billion training tokens.

¹⁷<https://huggingface.co/dbmdz/bert-base-italian-uncased>

¹⁸<https://github.com/idb-ita/GilBERTo>

¹⁹<https://github.com/musixmatchresearch/umberto>