ptt5-v2: A Closer Look at Continued Pretraining of T5 Models for the Portuguese Language

Marcos Piau[1][0009−0001−1490−3476], Roberto Lotufo[1,2][0000−0002−5652−0852], and Rodrigo Nogueira[1,3][0000−0002−2600−6035]

[1] School of Electrical and Computing Engineering, State University of Campinas (UNICAMP), Campinas, Brazil
[2] NeuralMind, Brazil
[3] Maritaca AI, Brazil

Abstract. Despite advancements in Natural Language Processing (NLP) and the growing availability of pretrained models, the English language remains the primary focus of model development. Continued pretraining on language-specific corpora provides a practical solution for adapting models to other languages. However, the impact of different pretraining settings on downstream tasks remains underexplored. This work introduces `ptt5-v2`, investigating the continued pretraining of T5 models for Portuguese. We first develop a baseline set of settings and pretrain models with sizes up to 3B parameters. Finetuning on three Portuguese downstream tasks (ASSIN2 STS, ASSIN2 RTE, and TweetSentBR) yields SOTA results on the latter two. We then explore the effects of different pretraining configurations, including quality filters, optimization strategies, and multi-epoch pretraining. Perhaps surprisingly, their impact remains subtle compared to our baseline. We release `ptt5-v2` pretrained checkpoints and finetuned MonoT5 rerankers on HuggingFace in their respective collections at https://huggingface.co/unicamp-dl.

Keywords: Portuguese T5 · Continued pretraining · Information Retrieval

1 Introduction

Transformer-based pretrained language models have established themselves as the core paradigm in the field of Natural Language Processing (NLP). Starting with the advent of BERT [12], which popularized the "pretrain, then fine-tune" approach and the use of the transformer architecture ifself, these models acquire a general-purpose language representation by unsupervised pretraining on extensive corpora of unlabeled text. The dynamics of the pretraining process had been studied in depth by many works, like Raffel et al. [34], that introduced T5, and scaled up models to billions of parameters and set new SOTAs across many tasks. The trend towards increasing model sizes and datasets to improve performance motivated studies like Kaplan et al. [23] on scaling laws and Hoffmann et al. [18], who demonstrated the importance of training data size relative to model size for compute-optimal training regimes; more recently work from Gadre et al. [14] specifically examined the influence of extended pretraining on downstream task performance.

Despite the extensive study of pretraining dynamics, the focus has predominantly been on English, leaving non-English languages less explored. Continued pretraining presents a strategic approach to adapting these models to additional languages and domains using significantly less data and computational resources than training from scratch. This method involves further pretraining on language-specific corpora, which has been shown to substantially enhance model performance on downstream tasks in the target language [32,6,25,5,42,11]. However, there is a lack of detailed investigations into how different settings during the continued pretraining phase influence on downstream tasks performance, with most studies merely aiming for benchmark-leading results without a thorough examination of the underlying factors.

In this work, we study the continued pretraining of T5 models for the Portuguese language, analyzing the impact of various settings on downstream task performance. Rather than solely focusing on achieving state-of-the-art results, our study also investigates how factors like model size, optimization schedules, and the application of quality filters in the pretraining dataset affect the performance. We continue the pretraning of Google's T5 with up to 3 billion parameters on Portuguese texts. By experimenting with different configurations in the pretraining stage, we observe nuanced effects on downstream tasks, with some settings only marginally outperforming the baselines. Our findings also suggest that while continued pretraining enhances model capabilities, the increments in performance diminish as model size increases.

T5 models [34] demonstrate adaptability across various natural language processing (NLP) tasks due to their encoder-decoder architecture. This structure enables them to process text for both understanding and generation, providing an advantage over encoder-only models like BERT. While not the focus of this work, T5's adaptability to instruction-based fine-tuning, as seen in FLAN-T5 [8], also enables effective zero-shot and few-shot applications. These factors, combined with the scarcity of Portuguese pretrained encoder-decoder models, motivate our choice to continue the investigation the T5 architecture in this study.

2 Related Work

The T5 model [34] is an encoder-decoder transformer, and one of its main innovations was to cast all tasks into a text-to-text format, allowing for a unified approach; scaling up models to 11B parameters, they consolidated the transfer learning approach, setting new SOTAs for GLUE [46], SuperGLUE [45], CNN/Daily Mail [17] benchmarks. It was pretrained using the "span corruption" objective over the C4 ("Colossal Clean Crawled Corpus") dataset, where random consecutive spans in the input are replaced by special mask tokens, and the model is trained to predict these corrupted tokens (models with up to 11B parameters). Building upon this foundation, mT5 [48] extended the T5 framework to multilingual settings, having been pretrained on mC4, a multilingual dataset covering 101 languages (models up to 13B parameters). PTT5 [6] further adapted T5 for Portuguese by continuing the pretraining of T5 models on the BrWac dataset [44]. This approach led to significant improvements in downstream Portuguese language tasks, which were further enhanced by a Portuguese tokenizer. For clarity, we'll refer to the work of Carmo et al. as `ptt5-v1`. Other notable international adaptations of T5/mT5 include it5 [40] (Italian), AfriTeVa [22] (low-resource African languages), AraT5 [29] (Arabic), and plT5 [7] (Polish).

Bertimbau [42], a popular adaptation of the BERT encoder model, remains influential within Portuguese language modeling. Others exploring encoder architectures include Albertina [37], DeBERTinha [5], the work of Gomes et al. [16] (which pretrains a Roberta model), and de Morais et al. [28]. Reflecting a broader trend, numerous recent Portuguese models prioritize decoder-only architectures, such as Sabiá [32], Glória [27], Bode [15], Cabrita [25], and Gervásio [39]. In the encoder-decoder space, the work of Carmo et al. (`ptt5-v1`) explored adapting T5 models for Portuguese. Beyond generic Portuguese models, several works specialize in custom domains: de Barros et al. [2] and BERTabaporu [10] were designed for Portuguese social media data, while Bertaú [13] focuses on financial language.

3 Methodology

This section describes the methodology for pretraining and evaluating our key experiments, covering the pretraining dataset, language-specific vocabulary, model architectures, optimization strategies, and finetuning and validation processes for downstream tasks.

3.1 Unsupervised continued pretraining

As the pretraining data, we utilized the Portuguese segment of the mC4 dataset (hereafter referred to as mC4-pt), comprising approximately 524 GB of uncompressed text across 169 million documents. This dataset is significantly larger than the one used for the pretraining of `ptt5-v1` models, which originated from the BrWac dataset [44] and consisted of around 15 GB of text from 7.4 million documents after preprocessing.

We adopted the Portuguese language vocabulary from ptt5-v1. This SentencePiece Unigram tokenizer [24], comprising 32,000 tokens, was trained over a corpus of 2 million documents from the Portuguese Wikipedia. This vocabulary shares the same number of tokens and control tokens as T5, facilitating the direct use of Google's model checkpoints.

As the pretraining objective, the span corruption task was employed, utilizing batches of 128 sequences of 512 tokens (65,536 tokens) - a methodology consistent with the baseline experiment by Raffel et al. [34]. Adafactor optimizer [41] with a constant learning rate of 0.001 and cross-entropy as loss was used during the entire pretraining process. Using these experimental settings, we started from Google's original checkpoints with sizes from `t5-small` (60M parameters) up to `t5-3B` (3B parameters), and performed a complete epoch of continued pretraining over the mC4-pt dataset. Considering these settings, a single epoch over the mC4-pt dataset comprises approximately 1,764,515 training steps and 116 billion training tokens. Additional pretraining experiments are detailed in Section 5.1.

Both pretraining and finetuning experiments utilized TPUv2-8 and TPUv3-8 devices, leveraging `t5` [34] and `seqio` [36] frameworks.

3.2 Supervised finetuning on downstream tasks

We assess the impact of our pretraining on three Portuguese language downstream tasks: ASSIN2 RTE, ASSIN2 STS, and TweetSentBR. The ASSIN2 dataset [31] provides two tasks: RTE (Recognizing Textual Entailment), which involves determining whether one sentence entails another, and STS (Semantic Textual Similarity), which quantifies the semantic similarity between sentence pairs on a 1-5 scale. The TweetSentBR dataset [4] is a sentiment analysis task for Brazilian Portuguese tweets, classifying them as positive, negative, or neutral. Tables 1 and 2 show further details and examples for each task.

Dataset	Task	Preferred metric	Train/validation/test	Random score	Possible outputs
ASSIN2 RTE	Binary classification	F1-macro	6,500/500/2,448	50	{Entailment, None}
ASSIN2 STS	Regression	Pearson	6,500/500/2,448	0	$[1, 5]$
TweetSentBR	Multiclass Classification (3)	F1-macro	11,525/1,281/1,982	32.4	{negativo, positivo, neutro}

Table 1: Downstream tasks datasets.

Dataset	Example inputs	Example targets
ASSIN2 RTE	`assin2_rte sentence1: Uma pessoa está escovando um gato sentence2: O pelo de um gato está sendo penteado por uma pessoa`	`Entailment`
ASSIN2 STS	`assin2_stsb sentence1: Uma mulher está cortando vegetais sentence2: Uma mulher está cortando brócolis`	`4.2`
TweetSentBR	`ttsbr_neg_pos_neu_sentiment_pt: adorando esse com dr dráuzio varela`	`positivo`

Table 2: Downstream tasks inputs and targets

We finetuned the pretrained models over 100 epochs with batches of 128 sequences and a maximum length of 512 tokens, using Adafactor as the optimizer with a constant learning rate of 0.001. The model checkpoint yielding the best performance on the validation set was selected for testing, and greedy decoding was utilized as the decoding method. Because TweetSentBR lacks a validation set, we reserved 10% of the training data for validation and used the remaining 90% for training.

All tasks were approached using a text-to-text format. Specifically for the ASSIN2 STS task, which involves the prediction of continuous values in the range between 1 to 5, we adopted the strategy from Raffel et al. [34], by rounding the target scores to the nearest 0.2 increment and converting these to strings, thus framing it as a multiclass classification problem compatible with the text-to-text format.

To compare the quality of the new checkpoints against existing alternatives, we also use the same finetuning procedure on Google T5 and mT5 models.

3.3 MonoPTT5 Rerankers

To evaluate the adaptability of the ptt5-v2 models for information retrieval tasks, specifically passage reranking, we trained MonoT5 rerankers [30] using checkpoints generated as described in Section 3.1. We named these models MonoPTT5. MonoT5 rerankers are used for passage reranking, a two-stage process: first, a less computationally expensive method like BM25 generates an initial set of relevant documents for a given query; the reranker model then reranks a subset of these documents to improve relevance ordering. During training, the model learns in a supervised text-to-text manner to generate tokens corresponding to relevant and non-relevant labels. For inference, we greedy decode a single token and calculate the softmax over the logits of the two possible tokens, using the probability of the positive class as the relevance score.

We adapted the input and target format for the Portuguese language to the structure `"Pergunta: {query} Documento: {document} Relevante:"`, assigning the tokens "Sim" (relevant) and "Não" (non-relevant). This format is applied during both training and inference, regardless of the input language.

The training data originated from the mMARCO dataset [3], a translated version of the MS MARCO passage retrieval dataset [1], originally in English, to 13 languages, including Portuguese. The training subset consists of triples (query, relevant passage, non-relevant passage), which we split into two training example pairs, with each pair containing the query matched to one passage – either relevant or non-relevant – thus creating one example for each label. We created a bilingual Portuguese-English training dataset by randomly assigning one of the two languages to each training triplet. This "translate-train" approach [9,48,20] leverages synthetic data augmentation by integrating machine translations with original text data to substantially expand the available training material in the target language. Prior research [3,38,9] has shown the effectiveness of this bilingual training strategy, motivating our adoption of this method.

The models were trained for 100k steps with batch sizes of 128 sequences and a maximum length of 512 tokens, utilizing Adafactor with a constant learning rate of 0.001 as the optimizer. Instances exceeding the maximum token length were excluded from the training dataset; these constituted approximately 0.01% of the training samples and were predominantly attributed to noisy translation data. Given the significant computational resources required for training these rerankers, we focused exclusively on models based on the main `ptt5-v2` checkpoints.

To evaluate the rerankers, we first used BM25 to generate an initial set of relevant documents [4] and then rerank the top-1000 documents. Retrieval metrics are calculated by comparing this ordered list with the relevance judgments from

[4] All applications of BM25 in this work use Pyserini's implementation [26] with default parameters $k_1 = 0.9$ and $b = 0.4$

each dataset. We consider two retrieval scenarios: in-domain (using the "small dev" set of 6,980 queries from the mMARCO-pt dataset) and zero-shot (using the Portuguese subset of 249 annotated queries from mRobust[21]). Due to the longer document length in mRobust, we segmented documents into sliding sentence windows using a Spacy [19] sentencizer pipeline, with a maximum length of 8 and a stride of 4 sentences to mitigate truncation during reranking.

4 Main Results

Table 3 shows the results on the downstream tasks considered. In the ASSIN2 RTE task, our 3B sets a new SOTA, surpassing the current one by 0.61 F1-macro points. For the TweetSentBR dataset, we achieved better performance than current finetuned SOTAs with `ptt5-v2-large` and `ptt5-v2-3B`, by 0.52 and 1.54 F1-macro points, respectively, but our results are worse when comparing to GPT-4. We highlight that our `ptt5-v2` were trained exclusively on each task training data using the text-to-text framework without any data augmentation or adaptation to the model's architecture, unlike the works of [2] and [38], which held the SOTA for TweetSentBR and ASSIN2 RTE. In the ASSIN2 STS task, our models did not surpass the current SOTA; regardless, `ptt5-v2` still shows better performance than mT5 and T5 models with approximate sizes, and this is also the only task where a smaller `ptt5-v2` model (`ptt5-v2-large`) has better performance of a large one `ptt5-v2-3B`.

		Binary classification	Regression	Multiclass classification	←	Retrieval	
		ASSIN2 RTE	ASSIN2 STS	TweetSentBR	←	mMARCO-pt	mRobust-pt
Model	Parameters	F1-macro	Pearson	F1-macro	NPM	RR@10	nDCG@20
T5							
t5-small	60M	83.66	0.738	62.14	61.71	-	-
t5-base	220M	85.80	0.764	65.43	65.63	-	-
t5-large	770M	88.91	0.790	68.22	69.95	-	-
t5-3B	3B	90.78	0.827	72.58	74.56	-	-
mT5							
mt5-small	300M	75.36	0.688	61.81	54.35	-	-
mt5-base	580M	79.39	0.749	70.76	63.46	-	-
mt5-large	1.2B	88.25	0.753	61.11	64.76	-	-
mt5-xl	3.7B	91.81	0.827	77.05	77.45	-	-
ptt5-v2							
ptt5-v2-small	60M	87.14	0.782	70.99	69.86	0.273	0.344
ptt5-v2-base	220M	88.36	0.814	73.20	72.82	**0.311**	0.384
ptt5-v2-large	770M	91.73	0.839	**76.78**	77.68	**0.315**	**0.462**
ptt5-v2-3B	3B	**92.68**	0.829	**77.80**	78.48	**0.332**	**0.512**
Supervised SOTA	-	92.07 [38]	**0.868** [37]	76.26 [2]	-	0.306 [38]	0.391 [21]
Few shot baselines from [32]							
GPT-4	-	90.96	0.776	82.40	-	-	-
GPT-3.5-turbo	-	88.28	0.664	74.39	-	-	-

Table 3: Main results after finetuning. For supervised SOTAs, model sizes are as follows: [38] (335M), [37] (900M), [2] (110M), [38], and [21] (580M). NPM results exclude retrieval tasks.

In addition to using individual metrics for each task, we also incorporate the Normalized Preferred Metric (NPM) [43] to facilitate the evaluation of the overall performance of a pretrained model across multiple tasks. The NPM normalizes a task's preferred metric (e.g., F1-macro for ASSIN2 RTE), assigning a value of 0 to represent random performance and 100 to denote maximum performance. Below is the equation used to calculating the NPM for a given model and set N of tasks:

$$\text{NPM} = \frac{100}{N} \sum_{i=1}^{N} \frac{[\text{raw preferred metric}]_i - [\text{random score}]_i}{[\text{maximum score}]_i - [\text{random score}]_i} \tag{1}$$

Given that MonoPTT5 rerankers were exclusively trained starting from `ptt5-v2` pretrained checkpoints, retrieval tasks were excluded from this evaluation. Therefore, we only considered ASSIN2 RTE, ASSIN2 STS, and TweetSentBR tasks. For each model, we calculate its aggregate performance by first determining the NPM for each task and then computing the average of these values.

Our `ptt5-v2` models have higher NPM values than T5 and mT5 models with considerably more parameters: for example, `ptt5-v2-base` is only surpassed by `t5-3B` ($\sim$13.6x larger) and `t5-xl` ($\sim$16.81x larger); this performance gap, however, is most pronounced in smaller models, narrowing as model size increases. A similar result was also observed by Xue et al. [48], which analyzed the performance of T5 and mT5 models peformance on the SQuAD benchmark [35], observing a peformance gap between `t5-small` and `t5-base` vs mT5 models of equivalent sizes, which is diminished starting from `t5-large`. This performance gap observed on smaller model sizes is advantageous when we consider environments constrained by computational resources, increasing the maximum attainable level of performance; additionally, a language-specific tokenizer reduces text splitting into fewer tokens, leading to lower latency and the potential to accommodate more text within the same maximum token context window. Interestingly, mT5 models tend to show lower NPM values, except in the 3 billion parameter range, where they slightly outperform T5 models.

For the retrieval tasks, our MonoPTT5 rerankers were able to set new SOTAs for both mMARCO-pt and mRobust-pt. For the mMARCO-pt dataset, models starting from `t5-base` size were able to surpass the current SOTA; the 3 billion parameters reranker obtained a gain of +0.026 points of MRR@10. In the mRobust-pt, our large and 3B rerankers surpassed the current SOTA by +0.071 and +0.121 in terms of nDGC@20, respectively. A more in-depth analysis of the retrieval tasks is conducted on section 5.2.

5 Ablations

5.1 Additional pretraining experiments

This sections includes pretraining experiments additional to those described in 3.1.

Comparison with ptt5-v1: Given the title of our work, a pertinent question arises: *How do `ptt5-v2` models compare with the work in `ptt5-v1`?* A few key differences exist in the pretraining of `ptt5-v1`. Notably, it utilized BrWac, a considerably smaller dataset, and a slightly different pretraining objective (denoising, where some input tokens are masked and the model is trained to predict the original text, rather than span corruption). Additionally, `ptt5-v1` employed models ranging from `t5-small` to `t5-large` with an Adafactor optimizer using a learning rate of 0.003 (threefold larger than our setting). `ptt5-v1` also explored both T5's original English vocabulary and a Portuguese language-specific tokenizer. In contrast, ptt5-v2 exclusively uses the latter.

To ensure a fair comparison, we finetuned the `ptt5-v1` checkpoints following the methodology outlined in section 3.2. Figure 1 presents the NPM values for both `ptt5-v1` and `ptt5-v2`, alongside comparisons to mT5 and T5 models. The data corroborates the enhancements achievable through monolingual pretraining on the target language, which is further augmented by employing a dedicated tokenizer. In comparison between the two `ptt5` iterations, a performance disparity favoring `ptt5-v2` is apparent for the `small` and `large` sizes, with the largest gap observed for `large`; however, models of the `base` size exhibit marginally superior performance in the `ptt5-v1` variant. Surprisingly, mT5's performance lags behind all other models, including the monolingual English T5, except in the 3B parameter range, where it closely matches `ptt5-v2` and surpasses T5.

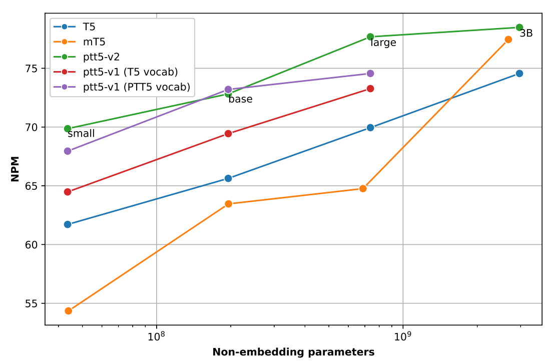

Fig. 1: NPM vs parameters for varying T5 configurations.

Quality filters: In our primary experiments, we utilized the entire mC4-pt dataset, containing approximately 116 billion training tokens, in the pretraining phase. In this additional experiment, we consider the MassiveText quality filters [33] to investigate the impact of this filtering process on downstream tasks. Applying these filters to mC4-pt reduces the number of training tokens to approximately 82 billion, a reduction of about 30%. This experiment was restricted to models of `t5-base` size, keeping the same batch size and optimization strategy used in the main experiments. Figure 2 shows the effect of using MassiveText's quality filters on downstream task performance, measured in terms of NPM. Pretraining with the filtered dataset shows an upward trend in performance, which continues without saturation up to the duration of one mC4-pt epoch, which is the last point in the plot. It's important to note that one epoch in the filtered dataset has fewer steps than the full dataset, so at one full dataset epoch, data is repeated for the filtered dataset. Despite the upward trend favoring the pretraining with the filtered dataset, the difference in terms of NPM is small at the one mC4-pt epoch mark.

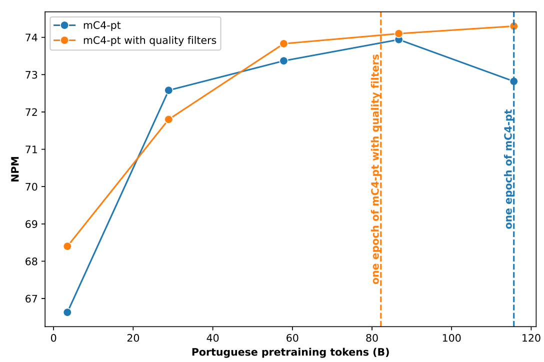

Fig. 2: Effect of using quality filters on mC4-pt. Vertical lines indicate the number of pretraining tokens for each dataset considered.

Pretraining optimization strategy: In our exploration of optimization strategies, we initially employed Adafactor with a constant learning rate. This ablation extends our investigation to the "inverse square root" learning rate schedule, as utilized by [34] in their final pretraining experiments. This learning rate schedule computes the rate as $\frac{1}{\sqrt{\max(n,k)}}$, where n represents the current step, and k is the number of warm-up steps. Raffel et al. [34] used $k = 10,000$, which sets the learning rate of 0.01 for the initial 10k steps, subsequently decreasing exponentially. The learning rate at the end of the pretraining, which consisted of around 1 million steps, was close to 0.001, the same one used during the finetuning. Figure 3 illustrates the difference between these optimization strategies.

Attempting to closely mirror the T5 pretraining recipe, we applied this identical schedule in our preliminary experiments. However, we observed a rapid overshoot in the training losses for `t5-large` `t5-3B` within hours; adjusting n only delayed the overshoot without averting it. Transitioning to a constant learning rate of 0.001 solved the overshoot issue, leading to stable pretraining loss across all model sizes and simplifying our experimental setup. Because overshoot was observed only in the larger models, and also given the computational costs associated with pretraining the larger models, we performed additional

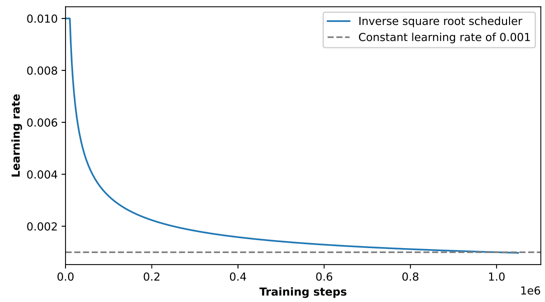

Fig. 3: Learning rate schedules: Constant vs. inverse square root scheduler as a function of training steps.

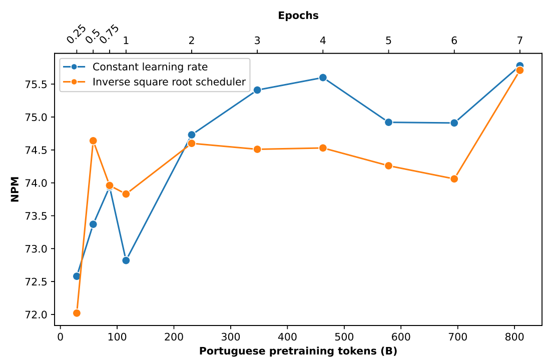

Fig. 4: Effect of scheduler used during pretraining. Epochs are relative to mC4-pt.

pretraining experiments with “inverse square root" scheduler for `t5-small` and `t5-base` models only.

Number of pretraining epochs: In the primary set of experiments, the mC4-pt dataset was fully utilized for pretraining over one epoch. To explore the influence of the number of pretraining epochs on downstream task performance, we conducted experiments over various epochs, including partial epochs (0.25, 0.5, and 0.75 of an epoch). Considering the significant time and computational resources required for extended pretraining, especially with larger models, we limited the pretraining of the `t5-large` model to two epochs and the `t5-3B` model to a single epoch. The reduced epoch duration for the `t5-small` and `t5-base` models enabled more extensive pretraining periods for these configurations.

In Figure 4, the NPM values for `t5-base` are shown with the constant and the inverse square root scheduler, across a varying number of pretraining epochs. It is observed that there is a difference between these two optimization strategies: the inverse square root scheduler has the advantage for up to two epochs; afterwards, the constant learning rate takes the upper hand, and by the last epoch considered, they reach the same value. An increasing trend in the values of NPM is also noted for more epochs.

5.2 MonoPTT5 Rerankers

The information retrieval tasks reported in 3 represents the performance of our MonoPTT5 experiments, developed with the methodology describe in 3.3; in this section we also report the results for other approaches, including BM25, and dense retrieval using `multilingual-e5` [47] models. The dense models are used as a single stage retrieval system without reranking; dense indexing and retrieval is performed with A100 and V100 GPUs on Google Colab, leveraging the Pyserini framework. For mRobust-pt, which contain longer documents, we mitigate document truncation by using the same splitting strategy described in the section 3.3, using the maximum score among the document segments as the document score.

Figures 5 and 6 are used to illustrate the discussion presented in this section. Results in Table 3 only shows the effectiveness in each retrieval task of our MonoPTT5 models and the SOTA competitors. For the in-domain retrieval task, the mMarco-pt dataset, we note that BM25 is easily surpassed by all alternatives considered, and the effectiveness figures for MonoPTT5 rerankers and `multilingual-e5` models are similar when consider the size range in common, and MonoPTT5 rerankers effectiveness is above SOTA starting from models of size `t5-base`. For mRobust-pt, representing a zero-shot setting, BM25 is only surpassed by the mT5 reranker from Jeronymo et al. [21], and MonoPTT5 models starting from `t5-base` size.

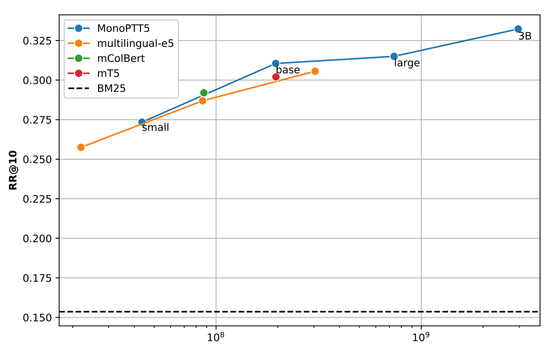

Fig. 5: Retrieval results on mMarco-pt. mColbert and mT5 values are from Bonifacio et al. [3]. Total size excludes embedding parameters.

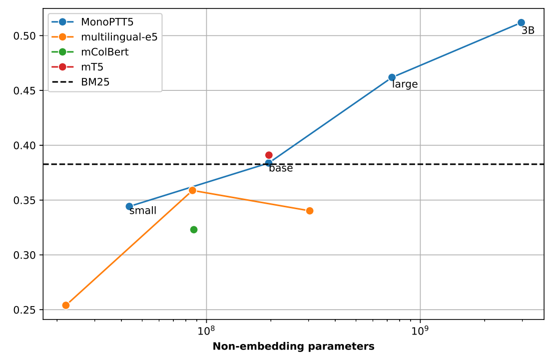

Fig. 6: Retrieval results on mRobust-pt. mColbert and mT5 values are from Jeronymo et al. [21]. Total size excludes embedding parameters.

6 Conclusion

In this study, we introduced `ptt5-v2`, exploring the continued pretraining of T5 models for the Portuguese language. We pretrained T5 models using a Portuguese language tokenizer, over a Portuguese language corpus. The finetuned models achieved SOTA on ASSIN2 RTE and TweetSentBr datasets, two of the three downstream tasks considered. Additionally, we applied these pretrained checkpoints to develop MonoT5 rerankers customized for the Portuguese language, achieving top performance on the mMARCO-pt and mRobust-pt datasets. Our main results supports the evidence of a performance gap favoring monolingual models over English-focused and multilingual models, a gap that narrows as model capacity increases. This underscores the importance of language-specific pretraining, and our analysis of pretraining settings suggests that while data filtering, optimization strategies, and pretraining duration can offer incremental improvements, the overall effects were limited in comparison to our baseline settings and the core pretraining recipe remained robust.

Acknowledgements

We thank Google for the TPU grant through the TRC program.

References

1. Bajaj, P., Campos, D., Craswell, N., Deng, L., Gao, J., Liu, X., et al.: Ms marco: A human generated machine reading comprehension dataset (2018)
2. de Barros, T.M., Pedrini, H., Dias, Z.: Leveraging emoji to improve sentiment classification of tweets. In: Proceedings of the 36th Annual ACM Symposium on Applied Computing. p. 845–852. SAC '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3412841.3441960

3. Bonifacio, L., Jeronymo, V., Abonizio, H.Q., Campiotti, I., Fadaee, M., Lotufo, R., et al.: mmarco: A multilingual version of the ms marco passage ranking dataset (2022)
4. Brum, H., Volpe Nunes, M.d.G.: Building a sentiment corpus of tweets in Brazilian Portuguese. In: Calzolari, N., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., Tokunaga, T. (eds.) Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). European Language Resources Association (ELRA), Miyazaki, Japan (May 2018)
5. Campiotti, I., Rodrigues, M., Albuquerque, Y., Azevedo, R., Andrade, A.: Debertinha: A multistep approach to adapt debertav3 xsmall for brazilian portuguese natural language processing task (2023)
6. Carmo, D., Piau, M., Campiotti, I., Nogueira, R., Lotufo, R.: Ptt5: Pretraining and validating the t5 model on brazilian portuguese data (2020)
7. Chrabrowa, A., Łukasz Dragan, Grzegorczyk, K., Kajtoch, D., Koszowski, M., Mroczkowski, R., et al.: Evaluation of transfer learning for polish with a text-to-text model (2022)
8. Chung, H.W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., et al.: Scaling instruction-finetuned language models (2022)
9. Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., et al.: Unsupervised cross-lingual representation learning at scale (2020)
10. Costa, P.B., Pavan, M.C., Santos, W.R., Silva, S.C., Paraboni, I.: BERTabaporu: Assessing a genre-specific language model for Portuguese NLP. In: Mitkov, R., Angelova, G. (eds.) Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing. pp. 217–223. INCOMA Ltd., Shoumen, Bulgaria, Varna, Bulgaria (Sep 2023)
11. Cui, Y., Yang, Z., Yao, X.: Efficient and effective text encoding for chinese llama and alpaca (2024)
12. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding (2019)
13. Finardi, P., Viegas, J.D., Ferreira, G.T., Mansano, A.F., Caridá, V.F.: Bertaú: Itaú bert for digital customer service (2021)
14. Gadre, S.Y., Smyrnis, G., Shankar, V., Gururangan, S., Wortsman, M., Shao, R., et al.: Language models scale reliably with over-training and on downstream tasks (2024)
15. Garcia, G.L., Paiola, P.H., Morelli, L.H., Candido, G., Júnior, A.C., Jodas, D.S., et al.: Introducing bode: A fine-tuned large language model for portuguese prompt-based task (2024)
16. Gomes, J.R.S., Garcia, E.A.S., Junior, A.F.B., Rodrigues, R.C., Silva, D.F.C., Maia, D.F., et al.: Deep learning brasil at absapt 2022: Portuguese transformer ensemble approaches (2023)
17. Hermann, K.M., Kočiský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., et al.: Teaching machines to read and comprehend (2015)
18. Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., et al.: Training compute-optimal large language models (2022)
19. Honnibal, M., Montani, I., Landeghem, S.V., Boyd, A.: spacy: Industrial-strength natural language processing in python. To appear (2020). https://doi.org/10.5281/zenodo.1212303
20. Hu, J., Ruder, S., Siddhant, A., Neubig, G., Firat, O., Johnson, M.: Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization (2020)

21. Jeronymo, V., Nascimento, M., Lotufo, R., Nogueira, R.: mrobust04: A multilingual version of the trec robust 2004 benchmark (2022)
22. Jude Ogundepo, O., Oladipo, A., Adeyemi, M., Ogueji, K., Lin, J.: AfriTeVA: Extending ?small data? pretraining approaches to sequence-to-sequence models. In: Cherry, C., Fan, A., Foster, G., Haffari, G.R., Khadivi, S., Peng, N.V., Ren, X., Shareghi, E., Swayamdipta, S. (eds.) Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing. pp. 126–135. Association for Computational Linguistics, Hybrid (Jul 2022). https://doi.org/10.18653/v1/2022.deeplo-1.14
23. Kaplan, J., McCandlish, S., Henighan, T., Brown, T.B., Chess, B., Child, R., et al.: Scaling laws for neural language models (2020)
24. Kudo, T., Richardson, J.: Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. arXiv preprint arXiv:1808.06226 (2018)
25. Larcher, C., Piau, M., Finardi, P., Gengo, P., Esposito, P., Caridá, V.: Cabrita: closing the gap for foreign languages (2023)
26. Lin, J., Ma, X., Lin, S.C., Yang, J.H., Pradeep, R., Nogueira, R.: Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In: Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021). pp. 2356–2362 (2021)
27. Lopes, R., Magalhães, J., Semedo, D.: Glória – a generative and open large language model for portuguese (2024)
28. de Morais, L.P., da Silva Soares, A., da CM Borges, V., da Silva, N.F.F., Pereira, F.S.: Sub-language sentiment analysis in whatsapp domain with deep learning approaches. Revista de Sistemas de Informaçao da FSMA **1**(31), 32–47 (2023)
29. Nagoudi, E.M.B., Elmadany, A., Abdul-Mageed, M.: Arat5: Text-to-text transformers for arabic language generation (2022)
30. Nogueira, R., Jiang, Z., Pradeep, R., Lin, J.: Document ranking with a pretrained sequence-to-sequence model. In: Cohn, T., He, Y., Liu, Y. (eds.) Findings of the Association for Computational Linguistics: EMNLP 2020. pp. 708–718. Association for Computational Linguistics, Online (Nov 2020). https://doi.org/10.18653/v1/2020.findings-emnlp.63
31. Oliveira, H.G., Real, L., Fonseca, E. (eds.): Proceedings of the ASSIN 2 Shared Task: Evaluating Semantic Textual Similarity and Textual Entailment in Portuguese, Extended Semantic Web Conference. No. 2583 in CEUR Workshop Proceedings (2020)
32. Pires, R., Abonizio, H., Almeida, T.S., Nogueira, R.: Sabiá: Portuguese Large Language Models, p. 226–240. Springer Nature Switzerland (2023). https://doi.org/10.1007/978-3-031-45392-2_15
33. Rae, J.W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., et al.: Scaling language models: Methods, analysis & insights from training gopher (2022)
34. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv preprint arXiv:1910.10683 (2019)
35. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: Squad: 100,000+ questions for machine comprehension of text (2016)
36. Roberts, A., Chung, H.W., Levskaya, A., Mishra, G., Bradbury, J., Andor, D., et al.: Scaling up models and data with `t5x` and `seqio` (2022)
37. Rodrigues, J., Gomes, L., Silva, J., Branco, A., Santos, R., Cardoso, H.L., et al.: Advancing neural encoding of portuguese with transformer albertina pt-* (2023)

38. Rosa, G.M., Bonifacio, L.H., de Souza, L.R., Lotufo, R., Nogueira, R.: A cost-benefit analysis of cross-lingual transfer methods (2021)
39. Santos, R., Silva, J., Gomes, L., Rodrigues, J., Branco, A.: Advancing generative ai for portuguese with open decoder gervásio pt* (2024)
40. Sarti, G., Nissim, M.: It5: Large-scale text-to-text pretraining for italian language understanding and generation (2022)
41. Shazeer, N., Stern, M.: Adafactor: Adaptive learning rates with sublinear memory cost (2018)
42. Souza, F., Nogueira, R., Lotufo, R.: Bertimbau: Pretrained bert models for brazilian portuguese. In: Cerri, R., Prati, R.C. (eds.) Intelligent Systems. pp. 403–417. Springer International Publishing, Cham (2020)
43. Srivastava, A., Rastogi, A., Rao, A., Shoeb, A.A.M., Abid, A., Fisch, A., et al.: Beyond the imitation game: Quantifying and extrapolating the capabilities of language models (2023)
44. Wagner Filho, J.A., Wilkens, R., Idiart, M., Villavicencio, A.: The brwac corpus: A new open resource for brazilian portuguese. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018) (2018)
45. Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., et al.: Superglue: A stickier benchmark for general-purpose language understanding systems (2020)
46. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: Glue: A multi-task benchmark and analysis platform for natural language understanding (2019)
47. Wang, L., Yang, N., Huang, X., Yang, L., Majumder, R., Wei, F.: Multilingual e5 text embeddings: A technical report (2024)
48. Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., et al.: mt5: A massively multilingual pre-trained text-to-text transformer (2021)