Introduction of mT5-LatinSummarizer, a version of mT5-small finetuned on Latin summarization

Axel Delaval, Elsa Lubek
École Polytechnique
Email: axel.delaval@gmail.com, elsa.lubek@gmail.com

GitHub Repository Hugging Face Model Hugging Face Dataset

***Abstract*—Despite the rapid advancements in natural language processing (NLP), low-resource languages remain significantly underrepresented in large-scale AI models. Latin, an ancient language with profound historical and academic importance, exemplifies this challenge due to its scarcity of digitized corpora and annotated datasets for NLP applications. In this study, we address the problem of Latin text summarization by fine-tuning a multilingual transformer-based model, mT5, specifically for a low-resource setting. We explore parallel corpus construction, data augmentation strategies (combining automatic extractive techniques and llms to produce summaries), and training methodologies such as curriculum learning or transfer learning tailored to limited computational resources. Furthermore, we discuss and compare different evaluation techniques specifically adapted to latin and latin summaries. Our new model, mT5-LatinSummarizer, is able to summarize Latin texts in Latin. Our findings demonstrate that pre-training a model with synthetic extractive summaries is crucial before fine-tuning it on translation-based summarization. This research underscores the importance of scalable approaches for low-resource NLP and provides insights into enhancing multilingual transformer models for historically and academically significant languages.**

***Index Terms*—Natural Language Processing, Low-Resource Languages, Machine Learning, Summarization, Latin, Neural Networks, mT5.**

Contents

1 Introduction

Low-resource languages present unique challenges for NLP, particularly in summarization tasks. Latin, a historically significant language, lacks large-scale datasets, making direct training difficult. This work explores strategies for Latin summarization by leveraging translation, extractive summarization, and curriculum learning.

In this paper, we do not aim at achieving the state of the art in matter of low-resource summarization, mostly because

of the lack of time and computational resources, but instead we aim to provide a scalable approach to tackle this difficult task.

Our work includes :

- Demonstration of the benefits of pre-training on a task that can be performed deterministically, showing how a **language model can develop useful internal representations beyond rule-based methods**.
- **Combination of extractive and abstractive summaries in training**, leveraging synthetic extractive summaries (SES) and Mistral-generated and translated abstractive summaries to enhance both grammatical accuracy and semantic richness.
- **Use of a large language model (LLM) to filter and refine training data**, improving summary quality by removing irrelevant or low-quality examples, even though it does not generate itself summaries.
- Empirical validation of the proposed method, demonstrating **promising results** in Latin summarization

1.1 Literature review

1.1.1 Challenges in Latin NLP: The development of NLP systems for Latin presents several constraints:

- **Scarcity of annotated data**: Unlike high-resource languages, Latin lacks large-scale summarization datasets. While some translated and untranslated corpora exist [1], the absence of human-annotated summaries makes supervised fine-tuning difficult. Furthermore, most of the translated corpora is not aligned.
- **Linguistic complexity and evolution**: Latin exhibits a highly inflectional morphology and flexible word order, which pose difficulties for NLP models trained on predominantly analytic languages. Furthermore, Latin has evolved over centuries, leading to variations across different historical periods, further complicating automatic processing.
- **Ineffectiveness of standard NLP metrics**: Traditional evaluation metrics such as BLEU and ROUGE, which rely on word-order similarity, often perform poorly for Latin due to its syntactic flexibility. More robust character-based or embedding-based metrics are needed to assess summarization quality accurately.

1.1.2 Linguistic properties of Latin: Latin differs from many contemporary languages in ways that significantly impact NLP processing:

- **Morphological complexity**: Unlike English, which relies on word order for syntactic structure, Latin heavily depends on inflectional morphology. Nouns, verbs, and adjectives change forms based on case, tense, mood, and other grammatical features, requiring NLP models to capture intricate morphological dependencies.
- **Relation to romance languages**: Latin serves as the root of Romance languages such as French and Italian, meaning that embeddings trained on these languages may offer some transfer learning benefits.
- **Historical variability**: Classical Latin, Medieval Latin, and Neo-Latin exhibit significant grammatical and lexical differences, making it difficult to train a model that generalizes well across all historical texts.

1.1.3 Limitations of existing Latin NLP tools: Although some NLP tools have been developed for Latin [1], many are outdated or lack compatibility with modern deep learning frameworks:

- **Classical Language Toolkit (CLTK)**: One of the few toolkits dedicated to historical languages, CLTK [2] provides lemmatization, part-of-speech tagging, and syntactic parsing for Latin. However, it is primarily rule-based and does not integrate well with deep learning libraries such as PyTorch or TensorFlow.
- **Latin-BERT**: A transformer-based model pre-trained on Latin, Latin-BERT could potentially enhance NLP applications [3]. However, its size and lack of compatibility with recent frameworks make it difficult to integrate into modern summarization pipelines.

Given these limitations, our approach focuses on adapting multilingual transformers like mT5 [4] to Latin NLP tasks, rather than relying on specialized Latin-specific models that may be difficult to fine-tune or scale.

2 Translation: Fine-Tuning mT5

Most large-scale language models (LLMs) provide minimal support for Latin, as it represents only a small fraction of their training data. Even state-of-the-art models such as GPT-4o or Claude struggle with Latin grammar, indicating an underrepresentation in mainstream NLP research. Given these constraints, we selected mT5-small[1] , a compact multilingual transformer model trained on the mC4 corpus, which explicitly includes Latin data [4] and fine-tuned it in order to obtain a mT5-LatinTranslator which can run on a small GPU (16 Go).

2.1 mT5-LatinTranslator

Despite this inclusion, Latin remains a minor component of mT5-small training set. To evaluate mT5-small's capabilities, we initially fine-tuned it for English-Latin translation using a dataset of 200,000 sentence pairs, creating a **mT5-LatinTranslator model**.

2.2 Fine-Tuning mT5 on Parallel Datasets

To enhance Latin summarization performance, we fine-tuned mT5-small using parallel English-Latin datasets. The following datasets[2] were used:

- HuggingFace Latin-English Translation Dataset (101,371 sentence pairs)
- Bible-uedin Opus and Wikimedia Opus (163,067 sentence pairs)

After filtering out duplicates and low-quality data, we retained **213,226 sentence pairs** for fine-tuning on a translation

[1] https://huggingface.co/google/mt5-small

[2] All the datasets used for training can be found on our HuggingFace repository https://huggingface.co/datasets/LatinNLP/LatinSummarizerDataset

task, the statistics on the number of tokens can be found in Table I.

Statistic	English	Latin
Sentence Count	213,226	213,226
Mean Length (tokens)	29.49	19.12
Standard Deviation	15.80	9.99
Maximum Length	100	80

TABLE I: Sentence Length Statistics for the English-Latin Dataset

2.3 Additional new aligned dataset

We "manually" created another aligned dataset using data from the cltk library (perseus library) that we used in our experiment of integrating stanza (mT5-LatinTranslator2).

In future works, we could try using this dataset (without stanza annotations) to enhance our mT5-LatinTranslator and expose it to classical latin, and not only biblical and modern latin.

2.4 Results of mT5-LatinTranslator and mT5-LatinTranslator2

After three epochs, performance remained suboptimal, with BLEU scores of BLEU(en→la) = 18.03 and BLEU(la→en) = 20.04, indicating that the model struggled with Latin syntax and morphology. Moreover, we found out that a chrF score should better fit latin language and also computed it. Indeed, the order of words is not much important, but rather the declension (which is a relationship between words and their declension or conjugaison, so more local). Some concrete examples are provided in annexe. However, three epochs is probably not enough to capture the essence of Latin since mT5's exposure to this language was very poor, nevertheless in order to construct our prototype, we used this checkpoint for further translations.

2.5 Integrating Stanza for Latin-Specific adaptation

Although mT5 provides basic Latin capabilities, we sought to improve its performance using Stanza, a neural NLP pipeline [5], to preprocess data and enrich prompts with linguistic cues. Indeed, we noted that mT5-LatinTranslator sometimes translated twice the same word. It also failed to capture complex grammatical structures. Furthermore, providing a clue on the expected order of words could avoid the production of translations that are correct but not match the official one because of words order (in which case the loss function would be misleading).

Sentences (in latin and english) were annotated with morphological information (POS Tagging), exposing the model to word categories (e.g., nouns, verbs, adjectives).

To leverage Stanza's annotations, we designed structured prompts in which tokens were paired with their syntactic roles. Prompts followed patterns such as:

```
<en> <with_stanza> word1 <NOUN>
word2 <VERB> ... <en> <en.la> <la>
<with_stanza>
```

When clues were included, the prompt structure added a highlighted syntactic breakdown before the target translation:

```
<clue> <NOUN><ADJ><VERB>... <clue>
```

These structured hints were designed to help the model develop a syntactic understanding of Latin, reducing hallucination effects in translations and improving word order adequation with the official translation.

At first, when we tried training only with stanza, the results were very encouraging : the BLEU score without special token was lower (around 10) but the ChrF higher than without stanza (33.60 when we removed tags). However, our model failed at transferring knowledge from annotated prompts to unannotated prompts.

To avoid over-reliance on syntactic annotations while maintaining linguistic diversity, we applied a controlled prompt distribution:

- 60% raw text prompts: Direct sentence pairs without linguistic tags.
- 40% Stanza-augmented prompts: POS-enriched prompts. Half of them expected POS in the answer. Half of them included clues.

Even with this controlled prompt distribution, the results were still poor for raw text prompts (scoring below 1 at BLEU score and around 10 at Chrf score, with and without Stanza).

Nevertheless, we can imagine that using a tokenizer that better captures lemmas and declensions such as latin-BERT could probably improve scores in a future.

Finaly, we decided to keep the previous version (mT5-LatinTranslator) for latin translation.

3 Extractive summaries: Dataset construction for a pre-training

To fine-tune an SLM for Latin summarization using a limited number of high-quality summaries, we ideally need an SLM trained exclusively on Latin. However, since no such model exists, we opted to generate synthetic summaries (that we called **S**ynthetic **E**xtractive **S**ummary) by selecting the k most important sentences. This allows us to pre-train mT5-small by combining translation training with extractive summarization, ensuring it learns effective summarization patterns before fine-tuning on high-quality summaries.

We will see in the analysis of the results why this pre-training part was crucial. The choice of pre-training on extractive summaries, where key sentences are directly selected from the original text rather than being generated from scratch, was motivated by the following reasons :

- The 5000 High Quality (HQ) summaries were not sufficient to teach our model to produce latin.
- The quality of latin language and grammar accuracy is preserved in extractive summaries.
- Extractive summaries are easier to evaluate so the loss function of our model would be more accurate and the convergence of weights eased.

3.1 Extraction algorithm

To generate synthetic extractive summaries, we implemented an algorithm that combines TF-IDF vectorization, Latent Semantic Analysis (LSA), and redundancy reduction techniques. This approach is tailored to handle Latin texts, ensuring semantic accuracy and conciseness while preserving the original context.

3.1.1 Key steps in the algorithm:

1) **Text Preprocessing**:
   - Parenthetical content is removed to focus on core ideas.
   - Sentences are tokenized to create discrete units for analysis.
2) **TF-IDF Vectorization**:
   - Sentences are represented using n-grams ($n \in \{1, 2, 3\}$) to capture both individual terms and multi-word expressions (MWEs). This choice is supported by research showing that MWEs improve semantic representation, especially in morphologically rich languages like Latin [6].
   - Sublinear TF scaling and smooth IDF are applied to reduce bias toward frequent terms and prevent zero values for unseen terms.
3) **Latent Semantic Analysis (LSA)**:
   - Dimensionality reduction via Truncated Singular Value Decomposition (SVD) identifies latent semantic topics within the text. LSA helps uncover key sentences associated with distinct concepts .
4) **Sentence Ranking**:
   - Sentences are scored based on their importance in the reduced semantic space, ensuring selection of the most representative ones.
5) **Redundancy Reduction**:
   - Cosine similarity is used to filter out semantically redundant sentences, with a high similarity threshold (0.85) ensuring diversity while retaining thematic overlap.
6) **Context Preservation**:
   - The first sentence is always included to maintain narrative coherence.

3.1.2 Justification of design choices:

- **N-grams**: The use of n-grams captures word order and multi-word semantics, which are crucial for Latin texts with complex morphology.
- **LSA**: This technique effectively identifies key topics across sentences, as demonstrated in summarization research [7].
- **Redundancy reduction**: Filtering redundant sentences ensures concise summaries without sacrificing informativeness, aligning with best practices in extractive summarization.

This method balances linguistic nuance and computational efficiency, making it suitable for creating a large dataset of latin summaries in a short amount of time ($\sim 2 \cdot 10^{-2}$ s per extraction).

Discussion : we wanted to keep full sentences to ensure a minimum of coherence. However, in latin, some sentences can be made of many propositions and it could be convenient to remove part of them. An idea could be to use syntactic trees after a first extraction of complete sentence to reduce the selected sentences when their length seem too long. Even though the computation of syntactic trees can be slow (as we saw during the computation of stanza pos taggings which lasts about 2 s per sentence), limiting its use to the particular case of the very long extracted sentences should not slow down the generation process too much.

3.2 Ensuring extractive summary quality

Additionally, we designed a prompt that enables Mistral to grade Latin summaries. While Mistral[3] cannot generate Latin text fluently[4], it can understand and evaluate it using few-shot prompting (See Table VII). This method allows us to clean our SES dataset from examples were the pitfalls could not be detected or avoided by simple algorithms. First, we used some prompt engineering. We tested this method on 5,000 extractive summaries and plotted the resulting quality score distribution, which revealed two distinct clusters. We ensured that the grading was consistent with human appreciation. We noticed that all summaries which received a bad grade were of poor quality.

Although it is faster to prompt for evaluation rather than generation, the rating was quite long which is why we included examples that did not go through this validation process in our training dataset of SES.

A similar evaluation prompt could be used to evaluate the SLM's summaries afterwards.

4 High-Quality Summaries with Large Language Models (Mistral)

To generate high-quality Latin summaries, we used English as a pivot language, leveraging Mistral to produce refined summaries.

4.1 Approach: Intermediate Language Translation

A common technique to improve NLP performance for low-resource languages is to introduce an intermediate, well-supported language. We apply this approach to Latin summarization using the following steps:

1) Translate the Latin text into a high-resource language such as English.
2) Perform summarization in the intermediate language.
3) Translate the summary back into Latin.

[3]We used specifically the 7B-Instruct model : https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3

[4]Moreover, even if it could, it is very slow and therefore not scalable. We tried to use Mistral (which is faster than Llama) to generate extractive summaries but each summary was created in about 5 s each. However, grading requires only a few output tokens (the grade) and can be done in less than 0.5 s per summary.

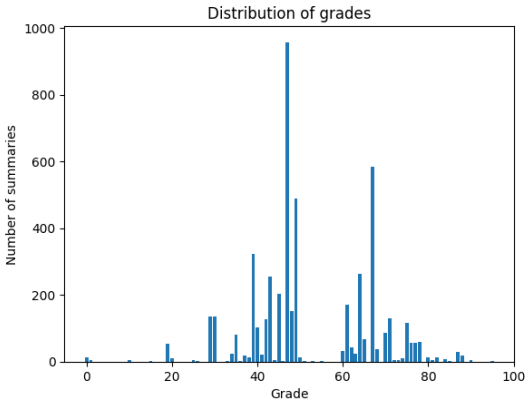

Fig. 1: Distribution of extractive summary quality scores assigned by Mistral.

A schema of the final pipeline is given in the annexe (see Figure 2).

This method takes advantage of better pre-trained summarization models in high-resource languages while mitigating the scarcity of direct Latin summarization data. We used our mT5-LatinTranslator model for the translation.

4.2 Justification for English as the Pivot Language

Choosing an appropriate pivot language is critical. Ideally, the intermediary language should be both well-supported in NLP and structurally similar to Latin. While languages like French or Italian share more linguistic similarities with Latin, English was chosen due to its significantly larger and higher-quality NLP datasets. Although we could have infer a French-Latin dataset from a English-Latin by using translation LLMs, the risk of losing semantic was too high.

An alternative approach would be to construct English-Latin and French-Latin parallel datasets and compare their effectiveness as pivot languages. However, due to computational constraints, we focused on English as the intermediary language. Future work could explore training and evaluating multiple pivot-language pipelines.

4.3 Generating Summarization Datasets with LLMs

Since no large-scale Latin summarization dataset exists, we leveraged modern large language models to generate synthetic training data. We used the **Mistral** model due to its efficiency (approximately 6 seconds per text) and ability to generate coherent summaries.

The following prompt was used to generate English summaries:

> [INST] Summarize the following text concisely: `{text}` [/INST]

To ensure high-quality training data, we applied length constraints and post-filtering techniques:

Future improvements could include human-in-the-loop evaluation to refine the dataset further and improve summarization quality.

Statistic	Length (tokens)
Sample Count	5,000
Mean Length	259.70
Standard Deviation	41.94
Minimum Length	200
Maximum Length	350

TABLE II: Latin Text Length Distribution in Generated Summarization Dataset

We could also compare in a future Mistral's English summaries translated by mT5-LatinTRanslator with summaries made directly in Latin by Mistral.

5 Experiments and results

In this part, we explain in detail our experiments and results. We also present our final model.

5.1 Experimental design

We also chose mT5-small for the summarization task, because of its pretraining on latin, and because a larger version of it (mT5) had already been identified as better in the context of summarization [8].

At first, we trained mT5-small on the High Quality (HQ) summaries. It did not produce any result because it was not able to produce latin answers from latin prompt.

Then, we took another mT5-small model and pre-trained it on repeating latin sentences (in addition to translations) before fineTuning it again on the HQ dataset (model mT5-LatinIdentity). The results were extremely poor again.

Finaly, we pre-trained another mT5-small model on our SES (synthetic extractive summaries) dataset and on translation tasks. It showed good results, already after the pre-training, and even better after the fine-tuning on abstractive summaries.

This pre-training on synthetic summaries can be interpreted as a form of curriculum learning, which has been shown to be even more efficient for low-resource language. The use of synthetic corpus during pre-training has already been done with hindi, but they used a synthetic corpus made by llm's which requires more computations [9].

The figure 2 shows the final optimal pipeline selected for our mT5-LatinTranslator model.

5.2 Evaluation methodology

Summaries evaluation is particularly difficult for a language such as latin. We used traditional ROUGE score, with the source texte and with the "gold summary", being here our HQ summary based on the translation method from Mistral english summary. Our "gold summaries" being not excellent due to our imperfect translator, we thought referring directly with the original text was pertinent. However, ROUGE score could favorise summaries that are too extractive and we also measured BERT-score, using a version fine-tuned for latin.

For a task such as summarization, what matters is more the content than the way it is phrased. That is why extractive summaries are much more easier to evaluate properly.

Model	Original Text	Generated Summary
mT5 fine-tuned on 4750 summaries after a pretraining on translation only	Quoniam comperi nonnullos, qui se plurimum sapere suis persuasionibus credunt, insanire, bacchari [...] exterminatos, statui pro captu ac mediocritate sermonis contraire [...]	<extra_id_0> aliquid aliquid aliquid aliquid aliquid aliquid [...]
mT5 fine-tuned on 4750 summaries after a pretraining on translation and "identity"	Daniel Günther Res apud repertae: Daniel Günther (natus die 24 Iulii 1973 Kieliae) rerum politicarum peritus Germanicus factionis CDU est. [...] Die 1 Novembris 2018 magistratum Praesidis Consilii Foederalis Germaniae iniit. [...]	., aea ecclesiae etiam natus) : CDU ; oeconomiae, eti us. etus iuee - etum genus ssibus rationes? eta ete

TABLE III: Comparison of mT5 fine-tuned models (on 7 epochs)

New techniques involve LLM as judges (as we did to filter our SES with Mistral).

GPT-4 based G-Eval establishes a new standard through chain-of-thought prompting, scoring summaries across coherence, consistency, fluency, and relevance. In multilingual evaluations, GPT-4 achieved 0.81 Spearman correlation with human Elo rankings, though performance dropped to 0.63 for Indonesian due to tokenization challenges. We can imagine that it is even worse for Latin although there has been no real recent study on that, which is why we preferred direct human appreciation. However, we included chat-GPT comments on our summaries in VII. The clear results even on a few examples were sufficient to conclude.

Another key feature of summaries is the density (number of characters of the summary over the one of the original text), which can especially affect recall. We computed density before and after fine-tuning and they were very similar (around 6%)

5.3 Comparison with and without Pre-training on Extractive summaries

As we can see on Table III, the results of the model without pretraining on extractive summaries are so poor that computing ROUGE score would not even make sense, which entice us to pre-train mT5 on an extractive summarization task before high-quality summaries.

5.4 Comparison between before and after finetuning

Table IV summarizes the evaluation metrics before and after fine-tuning[5].

Metric	Before Fine-tuning	After Fine-tuning
ROUGE-1	0.1675	**0.2541**
ROUGE-2	0.0427	**0.0773**
ROUGE-L	0.1459	**0.2139**
BERTScore-F1	0.6573	**0.7140**

TABLE IV: Comparison of evaluation metrics before and after fine-tuning (evaluated on the same 250 latin texts from Vicipaedia)

The results indicate that fine-tuning improved lexical overlap (ROUGE scores) and semantic similarity (BERTScore-F1), making the model more aligned with the input prompt.

[5]to compute these scores, we compared the generated summaries with the reference summaries

Another meaningful result is the Rouge-2 score when comparing the source text with the generated summary. Before fine-tuning, it obtains a Rouge-2 score of 0.096, meaning that a lot of digrams are reused in the generation (i.e. the summary is an extracted summary), but after the fine-tuning (100 epochs) this score goes down to 0.074.

5.5 Qualitative Comparison

To illustrate the improvements, we present a comparison of generated text before and after fine-tuning using selected prompts.

5.6 Observations

- The model after fine-tuning produces more coherent and semantically meaningful sentences.
- Before fine-tuning, outputs often contained incomplete or redundant phrases.
- The improvements in BERTScore-F1 indicate that the fine-tuned model better captures the meaning of the source text while remaining more fluent.
- The increase in ROUGE scores with the reference suggests better content retention from the source.
- The decrease in ROUGE-2 score with the original text prove less pure extraction.

Example after 100 epochs of finetuning :

> Text to summarize: Ager ubi hordeum colitur —— Plantae — Angiospermae Monocotyledones Commelinidae Ordo : Poales Familia : Poaceae Subfamilia : Pooideae Tribus : Hordeeae Genus : "Hordeum" Species : "Hordeum vulgare" Hordeum vulgare (binomen a Carolo Linnaeo anno 1753 statutum), Latinitate classica hordeum, est species generis "Hordei" et familiae Poacearum. [...] Catechin-7-O-glucosidum in "H. vulgare" inveni potest. Nexus externi.
>
> Generated Summary: Ager ubi hordeum colitur, species genus "Hordeo vulgare" Species a Carolo Linnaeo anno 1753 statutum, Latinitate classica Hordee, et familiae Poaceae.

We observe that the small language model (SLM) engages in frequent pure extraction, which can be attributed to the use of extractive summaries during pre-training. Despite this, it effectively identifies and extracts relevant information.

In our dataset, we deliberately excluded information enclosed in parentheses. However, the mT5-LatinTranslator retrieves such information regardless. Given that the extracted content remains relevant in this context, this behavior can be considered beneficial.

The SLM still produces grammatical errors. For instance, it incorrectly generates *"species generis Hordei"* instead of the grammatically correct *"species generis Hordeum"*, where *Hordeum* should be in the genitive case to properly indicate "species of the genus Hordeum." This issue is understandable, as the original text did not present the phrase within a complete sentence. Furthermore, even Mistral, which we used for high-quality summaries, when asked to answer in latin, exhibits similar errors.

Nevertheless, many of these grammatical mistakes could be corrected automatically using a grammar correction system, such as one built with Collatinus [2].

6 Training Configuration

6.1 Scalability Considerations

Given the limited time and computational resources available for this project, we prioritized designing a scalable prototype. This meant that at each stage of development, more epochs would significantly enhance the scores. Moreover, we generated more than one million extractive summaries of about 400 words from the 170M dataset but we used only less than a fifth.

6.2 Optimization Strategies

To maximize efficiency and minimize unnecessary computational overhead, several key optimizations were applied at different stages of training:

- **Gradient accumulation:** Since our batch sizes were constrained by GPU memory limitations, we implemented gradient accumulation to simulate larger batch sizes without exceeding memory capacity. Specifically, we set the accumulation steps to 1, allowing us to update gradients after multiple forward-backward passes.
- **Efficient data loading:** To reduce bottlenecks in data transfer between the CPU and GPU, we optimized the data pipeline using:
  - Prefetching to ensure data is loaded into memory before being needed.
  - Pinning memory to speed up data transfer from CPU to GPU.

  These techniques helped minimize training step latency.
- **Sampling the data:** Since the dataset is quite extensive for the resources we had to our disposal, at each epoch we sampled 10% of the training dataset so that each epoch was quicker.[6]
- **Dynamic sequence length adjustment:** Instead of processing inputs at a fixed maximum sequence length, we progressively increased the input sequence length during training. The formula used was very basic : $\texttt{max\_input\_length} = \min(\texttt{max\_seq\_len}\ ,\ \texttt{start\_epoch} \times 32)$ This approach acted as a form of curriculum learning, where the model started training on shorter sequences, allowing for larger batch sizes in the initial epochs. As training progressed, longer sequences were introduced, improving generalization without overwhelming GPU memory.
- **LoRA-based fine-tuning:** To efficiently adapt our pre-trained mT5 model to the task, we used Low-Rank Adaptation (LoRA), which introduced trainable low-rank matrices in select model layers. Specifically, we applied LoRA to the `query` and `value` matrices with the following configuration:
  - LoRA rank (r): 8
  - LoRA scaling factor (α): 32
  - LoRA dropout: 0.1

  This significantly reduced GPU memory consumption while maintaining model performance.
- **Learning rate scheduling:** A linear learning rate scheduler with warm-up was applied:
  - Learning rate: 5×10^{-4}
  - Warm-up steps: 1% of total training steps

  This prevented sudden spikes in loss and improved convergence stability.
- **Automatic Mixed Precision (AMP):** We implemented PyTorch's AMP but did not use it because it produced unstable losses. However, it would reduce memory consumption by dynamically using lower-precision floating-point calculations where applicable.

6.3 Infrastructure and Technical Challenges

Training was conducted via SSH connections to École Polytechnique's lab machines. However, several technical difficulties impeded our progress and required additional adaptations:

- **Hardware downtime:** A power outage over a weekend rendered the machines inaccessible, leading to a temporary halt in training.
- **Disk space limitations:** Checkpoints were not saved at times due to insufficient disk space.
- **GPU allocation constraints:** Queuing systems and GPU availability varied, sometimes leading to the impossibility of running LLMs (with our batch size and configuration, Mistral and mT5-small required 16Go of VRAM).

7 Conclusion and Future Work

7.1 Conclusion

Our training of mT5-LatinSummarizer using automatic extractive summary generation has yielded promising results. While some may question the value of pre-training a language model on a task that can be performed deterministically, our approach offers several advantages. First, the model's architecture is fundamentally different from rule-based algorithms, allowing it to develop novel internal representations

[6] This was important to keep frequent checkpoints since the ssh connection was not very stable (we encountered several power outages, etc, that interrupted the training loop)

that accelerate future learning. Moreover, leveraging a large language model (LLM) to evaluate and filter the generated corpus ensures higher-quality training data by discarding irrelevant summaries, a process that is challenging to automate effectively. Additionally, by training mT5-LatinSummarizer on both synthetic extractive summaries (SES) and abstractive summaries produced by Mistral, we aimed to achieve the best of both worlds: reduced grammatical errors and improved semantic richness.

Concerning transfer learning, we noticed that asking too many distinct tasks to our SLM (translations using or not stanza, using or not clues, and summarizations) did not produce satisfying results. We thus share an observation made in [8]: a multilingual SLM fine-tuned for non-english summarization performs better without transfer learning from english or multitask training. SLM's might be too small for the abstraction required for transfer learning (between tasks or languages).

Curriculum learning seems indeed to be a more effective approach (in terms of time and computational resources) than transfer learning, underlying the difficulties of language model to generalize quickly.

The originality of our work was to **not only propose a curriculum based on the difficulty of the task but also on an increasing quality of data, because training on high-quality data is not as much important in early stages**.

7.2 Future works and directions

Potential future work includes the following areas:

7.2.1 Immediate Enhancements and Experiments:

- **Improving the automatic generation of summaries** to enhance both syntactic accuracy and semantic coherence.
- **Utilizing LatinBERT** (once an up-to-date version is available, and potentially after quantization) to optimize tokenization and contextual embeddings.
- **Leveraging our newly aligned dataset** for translation tasks to improve bilingual performance.
- **Fine-tuning the mT5-LatinTranslator on a paragraph-aligned corpus**, now that the model demonstrates a fundamental understanding of Latin.
- **Determining the optimal ratio of content sources** by evaluating the balance between automatically generated content (filtered by humans or LLMs), human-authored content, and LLM-generated content.
- **Exploring adaptive learning rate strategies** to optimize model performance across different stages of pre-training and fine-tuning.
- **Investigating transfer learning approaches**, such as employing an alternative pivot language or incorporating prompts that request summaries in Romance languages (or English) to enhance Latin summarization capabilities. Our experiments with stanza-based transfer learning suggest that achieving meaningful improvements requires substantial additional training, which led us to exclude this approach from our initial study.

7.2.2 Scaling the Process:

- **Expanding pre-training with automatically generated corpora** to address Latin's low-resource status. The effectiveness of this approach suggests that it could be scaled further. For example, the "Collatinus automatic decliner" tool, previously available in the CLTK library, was not operational at the time of this study (March 2025) due to dependencies on other libraries. When it becomes available again, it could be leveraged to generate large quantities of grammatically correct, simplified Latin sentences. This could facilitate the creation of aligned English-Latin sentence pairs from children's literature, which, while not producing fully natural translations, could significantly expand the training corpus. Additionally, this tool could contribute to developing a **grammar correction module** to rectify declension errors and refine our synthetic extractive summary (SES) dataset.
- **Integrating a grammar correction system** directly into the training process to enhance the quality of both input and output text.

7.2.3 Application of Innovative Techniques:

- While this study focused on fine-tuning a pretrained small language model (SLM), future work could explore **modifying the underlying model architecture** to better accommodate Latin's linguistic complexity [10].
- **Tree-based architectures** could be particularly beneficial for representing Latin's hierarchical syntactic dependencies and flexible word order. Such non-linear structures more naturally align with the intricate grammatical relationships in Latin, as explored in other contexts (e.g., tabular data generation [11]).
- **Chain-of-Thought (CoT) reasoning techniques**—which have demonstrated effectiveness in solving sequential problems [12]—could be adapted to decompose complex Latin grammatical structures into logical steps, facilitating the processing of cases, genders, and verb conjugations.
- **Hybrid models combining transformers with specialized non-linear architectures** could further enhance the model's capacity to capture long-range grammatical dependencies in Latin text.

7.2.4 Exploration of Curriculum Learning (CL):

- **Structuring the training corpus chronologically (or inversely chronologically)**, starting from a Romance language and progressively transitioning to Classical Latin.
- **Incorporating metadata on the text's historical period or linguistic variety** to guide the model's adaptation. The classification of Latin variants could itself be performed through an external LLM or a dedicated algorithm, integrating grammatical, lexical, and contextual features.

ANNEXES: METHODS

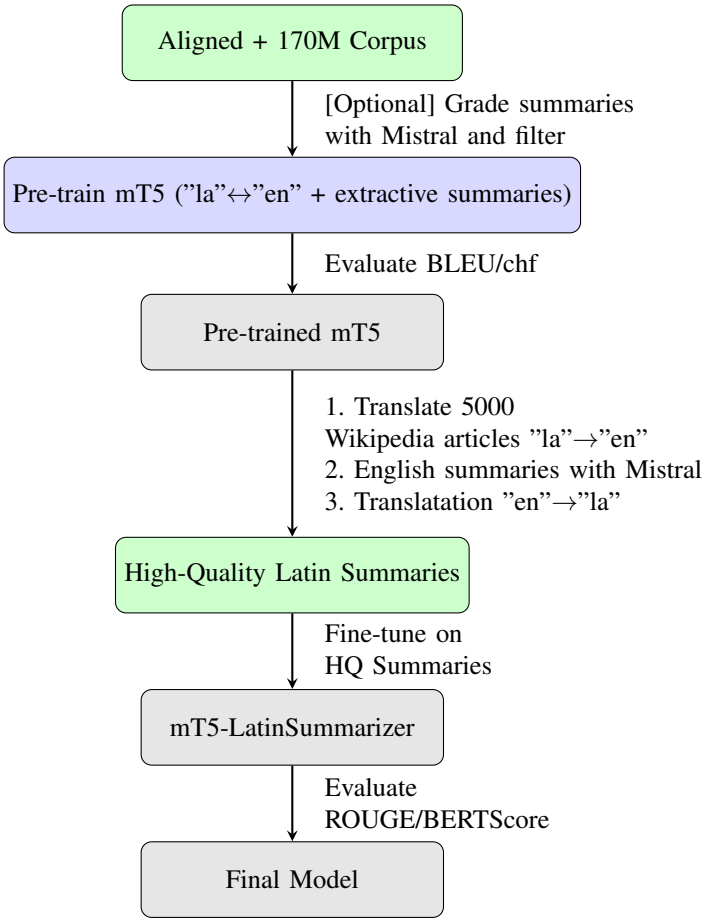

Fig. 2: Schematic representation of the pipeline for mT5-LatinSummarizer

English Input	**Latin Translation**
The sun is shining brightly today.	splendet sol hodie splendebatur lux
I love learning new languages.	meam didici linguis novi novi linguas
This book is very interesting.	Très fortissime libellus est valde.
Where is the nearest library?	Where is the nearest library?
He studied philosophy in Rome.	Philosophum Romae studuit.

TABLE V: English to Latin Translations

Latin Input	**English Translation**
Carpe diem.	Carpe diem.
Veni, vidi, vici.	"Come, see, cliffe".
Hodie pulchra tempestas est.	This became a very tempting during the summer season.
Librum legere amo.	I desire a book to read.

TABLE VI: Latin to English Translations

Original Text (Excerpt)	Summary	Grade	Explanation
Public Libraries *"Bibliotheca publica est bibliotheca quae, omnibus lectoribus patens, pecuniis publicis sicut vectigalibus plerumque exercetur. Institutio est quam operantur bibliothecarii operariique professionales qui etiam servi civiles sunt. Bibliothecis publicis sunt quinque proprietates fundamentales [...] Bibliotheca Malatestiana, prima bibliotheca a civitate operata, Caesenae in Italia anno 1447 constituta [...]"*	*"Bibliothecis publicis sunt quinque proprietates fundamentales: vectigalibus sustinentur; a consilio administrantur; omnibus patent; omnino voluntariae sunt; atque officia fundamentalia sine impensá praebent."*	53	The summary captures the main idea of public libraries but lacks depth. It fails to convey the historical progression and societal impact, making it a poor representation of the original. Additionally, the logical flow is weak—sentences jump between broad descriptions and specific details without smooth transitions. This disrupts readability and makes it harder to understand how public libraries evolved.
Erhard Hübener *"Erhardus Fridericus Iulius Hübener (natus 4 Augusti 1881 in vico Tacken, mortuus 3 Iunii 1958) vir publicus Germaniae et sodalis primo DDP, deinde LDPD fuit. Pater eius pastor protestans fuit. Anno 1901 maturitatem adeptus est et deinde Kieliae et Berolini oeconomiae studebat [...] Praeterea hoc tempore nova dictatura SED in oriente parte Germaniae orta est [...]"*	*"Pater eius pastor protestans fuit. Deinde mercator laborabat et primo bello mundano dux militum interfuit. Ab anno 1919 in administerio Borussiae commercii minister publicus laboravit."*	29	The summary is highly fragmented. While it mentions that Hübener was a German politician, it does not properly connect the key moments of his career. The chronological order is unclear, [...] Important transitions are missing, and some facts appear in isolation, reducing coherence.
...	...	...	...

TABLE VII: Extracts from the 5 few-shot examples given to Mistral for evaluation of summaries. The prompt looks like the following : *"[INST] You are an expert evaluator of extractive summaries [...] Scoring Guidelines : 90-100: A rare, flawless summary. [...] {few shot examples} [...]"*

REFERENCES

[1] S. et al, "Overview of the evalatin 2024 evaluation campaign," 2024. [Online]. Available: https://aclanthology.org/2024.lt4hala-1.21.pdf

[2] K. P. Johnson, P. J. Burns, J. Stewart, T. Cook, C. Besnier, and W. J. B. Mattingly, "The Classical Language Toolkit: An NLP framework for pre-modern languages," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, H. Ji, J. C. Park, and R. Xia, Eds. Online: Association for Computational Linguistics, Aug. 2021, pp. 20–29. [Online]. Available: https://aclanthology.org/2021.acl-demo.3/

[3] D. Bamman and P. J. Burns, "Latin bert: A contextual language model for classical philology," 2020. [Online]. Available: https://arxiv.org/abs/2009.10053

[4] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel, "mt5: A massively multilingual pre-trained text-to-text transformer," *arXiv preprint arXiv:2010.11934*, 2020.

[5] P. Qi, Y. Zhang, Y. Zhang, J. Bolton, and C. D. Manning, "Stanza: A python natural language processing toolkit for many human languages," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, A. Celikyilmaz and T.-H. Wen, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 101–108. [Online]. Available: https://aclanthology.org/2020.acl-demos.14/

[6] M. HÃ¼ning and B. SchlÃ¼cker, *24. Multi-word expressions*. Berlin, MÃ¼nchen, Boston: De Gruyter Mouton, 2015, pp. 450–467. [Online]. Available: https://doi.org/10.1515/9783110246254-026

[7] M. G. Ozsoy, "Lsa for text summarization using latent semantic analysis," *Journal of Information Science 37(4):405-417*, 2011.

[8] R. Calizzano, M. Ostendorff, Q. Ruan, and G. Rehm, "Generating extended and multilingual summaries with pre-trained transformers," in *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, N. Calzolari, F. Béchet, P. Blache, K. Choukri, C. Cieri, T. Declerck, S. Goggi, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, J. Odijk, and S. Piperidis, Eds. Marseille, France: European Language Resources Association, Jun. 2022, pp. 1640–1650. [Online]. Available: https://aclanthology.org/2022.lrec-1.175/

[9] R. Joshi, K. Singla, A. Kamath, R. Kalani, R. Paul, U. Vaidya, S. S. Chauhan, N. Wartikar, and E. Long, "Adapting multilingual LLMs to low-resource languages using continued pre-training and synthetic corpus: A case study for Hindi LLMs," in *Proceedings of the First Workshop on Natural Language Processing for Indo-Aryan and Dravidian Languages*, R. Weerasinghe, I. Anuradha, and D. Sumanathilaka, Eds. Abu Dhabi: Association for Computational Linguistics, Jan. 2025, pp. 50–57. [Online]. Available: https://aclanthology.org/2025.indonlp-1.6/

[10] A. Keersmaekers and W. Mercelis, "Adapting transformer models to morphological tagging of two highly inflectional languages: a case study on Ancient Greek and Latin," in *Proceedings of the 1st Workshop on Machine Learning for Ancient Languages (ML4AL 2024)*, J. Pavlopoulos, T. Sommerschield, Y. Assael, S. Gordin, K. Cho, M. Passarotti, R. Sprugnoli, Y. Liu, B. Li, and A. Anderson, Eds. Hybrid in Bangkok, Thailand and online: Association for Computational Linguistics, Aug. 2024, pp. 165–176. [Online]. Available: https://aclanthology.org/2024.ml4al-1.17/

[11] J. Li, B. Zhao, Z. Zhao, K. Yee, U. Javaid, and B. Sikdar, "Tabtreeformer: Tabular data generation using hybrid tree-transformer," 2025. [Online]. Available: https://arxiv.org/abs/2501.01216

[12] Z. Li, H. Liu, D. Zhou, and T. Ma, "Chain of thought empowers transformers to solve inherently serial problems," *arXiv preprint arXiv:2402.12875*, vol. 1, 2024.