Language Representation Models for Low- and Medium-Resource Languages

by

Jón Friðrik Daðason

Dissertation submitted to the Department of Computer Science
at Reykjavík University in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

March 2025

Thesis Committee:

Hrafn Loftsson, Supervisor
Associate Professor, Reykjavík University, Iceland

Sampo Pyysalo, Committee Member
Research Fellow, University of Turku, Finland

Anders Søgaard, Committee Member
Professor, University of Copenhagen, Denmark

Barbara Plank, Examiner
Professor, LMU Munich, Germany

ISBN 978-9935-539-78-6 (Print version)
ISBN 978-9935-539-79-3 (Electronic version)
ORCID: 0009-0009-5817-674X

Language Representation Models for Low- and Medium-Resource Languages

Jón Friðrik Daðason

March 2025

Abstract

Transformer-based language models have proven to be extremely effective for a wide variety of natural language understanding tasks, including question answering, automatic text summarization, and sentiment analysis. These models are typically pre-trained on large, unannotated corpora using self-supervised tasks such as masked token prediction, often requiring weeks or months of training, followed by fine-tuning on practical tasks, which requires substantially less time and data by comparison.

Since their introduction, Transformer models have grown exponentially in size, from approximately 100 million parameters in 2018 to over 600 billion in 2024, with the largest pre-training corpora growing from around 800 million tokens to over 14.8 trillion. However, many low- and medium-resource languages lack the extensive datasets and computational resources required to pre-train language models at this scale. Therefore, data-efficient pre-training techniques are crucial for effectively utilizing the limited resources available for these languages.

In this thesis, we investigate various data-efficient pre-training strategies and evaluate their impact on downstream tasks in six low- to medium-resource languages: Icelandic, Estonian, Basque, Galician, Nepali, and Tajik. First, we analyze several text quality filtering techniques to discard noisy data from web-crawled corpora. We propose a novel, language-independent filtering approach using unsupervised clustering and outlier detection algorithms which achieves comparable performance to a rule-based approach.

Second, we explore the effects of augmenting monolingual pre-training corpora with text from related and unrelated languages, as well as Python code, finding significant improvements in downstream performance for certain tasks for larger models. Our results support the hypothesis that linguistic similarity facilitates cross-lingual transfer.

Finally, we compare several subword tokenization algorithms and evaluate their impact on downstream results when used in pre-trained language models. Our analysis reveals that the Unigram algorithm consistently yields the best results on downstream tasks, and that a vocabulary size of 64k outperforms smaller vocabularies by a statistically significant margin.

Our findings demonstrate that data-efficient pre-training techniques can substantially improve the performance of language models for low- and medium-resource languages. By optimizing the use of available data and resources, we achieve statistically signifi-

cant improvements in downstream tasks under data-constrained conditions, paving the way for more effective natural language processing in resource-constrained settings.

We release several datasets and tools compiled and developed during the work of this thesis, as well as multiple pre-trained Transformer-based language models.

Keywords: Natural language processing, language models, text filtering, multilingual models, tokenization

Mállíkön fyrir tungumál með takmörkuð málföng.

Jón Friðrik Daðason

mars 2025

Útdráttur

Mállíkön sem byggja á Transformer-tauganetum hafa náð betri árangri en áður hefur þekkst fyrir máltækniverkefni eins og spurningasvörun, viðhorfsgreiningu og sjálfvirka samantekt. Slík líkön eru yfirleitt forþjálfuð á stórum, ómörkuðum málheildum á verkefnum eins og að endurheimta falin orð út frá samhengi, en sú þjálfun getur staðið yfir í margar vikur eða jafnvel mánuði. Því næst er hægt að fínstilla líkönin fyrir hagnýtari verkefni sem krefst mun minni tíma og þjálfunargagna.

Stærð Transformer-líkana hefur aukist gífurlega síðan þau voru fyrst kynnt til sögunnar, en þau hafa stækkað úr u.þ.b. 100 milljón stikum árið 2018 í yfir 600 milljarða stika árið 2024. Á sama tímabili hafa stærstu forþjálfunarmálheildirnar stækkað úr 800 milljónum orða í 14.800 milljarða. Mörg tungumál skortir bæði þjálfunargögn og reikniafl til að raunhæft sé að forþjálfa líkön af þessari stærðargráðu. Skilvirkar forþjálfunaraðferðir skipta því sköpum til að nýta takmörkuð málföng fyrir þessi tungumál á sem bestan hátt.

Í þessari ritgerð rannsökum við ýmsar mismunandi forþjálfunaraðferðir og metum skilvirkni þeirra á máltækniverkefnum fyrir sex tungumál þar sem málföng eru af skornum skammti: íslensku, eistnesku, basknesku, galisísku, nepölsku, og tadsísku. Í fyrsta lagi berum við saman ýmsar aðferðir til að sía ótæka texta úr vefmálheildum. Við lýsum nýjum textasíunarflokkurum sem byggja á reikniritum fyrir klösun og útlagagreiningu (e. outlier detection), sem ná sambærilegum árangri og aðferðir sem byggja á reglum.

Í öðru lagi metum við áhrifin af því að bæta texta úr skyldum og óskyldum tungumálum, auk Python forritunarkóða, við einmála forþjálfunarmálheildir. Tilraunir okkar gefa til kynna að fyrir stærri líkön geti þetta leitt til betri árangurs fyrir sum verkefni. Þessar niðurstöður renna stoðum undir þá tilgátu að forþjálfuð mállíkön eigi auðveldara með að yfirfæra þekkingu á milli skyldra tungumála.

Að lokum berum við saman mismunandi reiknirit fyrir tilreiðara og metum áhrif þeirra á niðurstöður úr máltækniverkefnum þegar þau eru notuð í forþjálfuðum mállíkönum. Tilraunir okkar sýna að Unigram reikniritið skilar bestum árangri og að það fást marktækt betri niðurstöður með 64k orðaforða heldur en með minni stærðum.

Niðurstöður okkar sýna fram á að skilvirkar forþjálfunaraðferðir geti bætt árangur fyrir tungumál þar sem málföng eru af skornum skammti. Með betri nýtingu á tiltækum þjálfunargögnum og reikniafli náum við fram marktækt betri niðurstöðum á máltækniverkefnum.

Við gefum út nokkur gagnasöfn og tól sem voru sett saman og þróuð við vinnu þessarar ritgerðar, auk fjölda forþjálfaðra mállíkana.

Efnisorð: Máltækni, mállíkön, textasíun, margmála mállíkön, tilreiðing

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Hrafn Loftsson, for his unwavering support, insightful guidance, and encouragement throughout the course of my PhD. His expertise and mentorship have been invaluable in shaping my research.

I am also sincerely thankful to the members of my PhD committee, Anders Søgaard, Professor at the University of Copenhagen, and Sampo Pyysalo, Research Fellow at the University of Turku, for their valuable feedback and constructive suggestions, which have significantly strengthened the quality of this work.

My journey in natural language processing began in the summer of 2010 at the Árni Magnússon Institute for Icelandic Studies, where I developed the spelling correction tool Skrambi. With the continued guidance of Sven Þ. Sigurðsson, Kristín Bjarnadóttir, and Sigrún Helgadóttir, I further developed this work in the years that followed, making it the focus of my Master's thesis. I consider myself truly fortunate to have had their enduring support and mentorship, and extend to them my heartfelt thanks.

Finally, I am profoundly grateful to my family for their unwavering support, patience, and encouragement throughout this journey.

The work presented in this thesis was funded by:

- The Icelandic Strategic Research and Development Program for Icelandic, 2019–2020, grant no. 180037-5301. "Automatic Text Summarization for Icelandic".
- Almannarómur: Language Technology Programme for Icelandic 2019–2023.
- Reykjavik University Research Fund, 2023.

This research was also supported with Cloud TPUs from Google's TPU Research Cloud (TRC).

Contents

List of Figures

List of Tables

List of Abbreviations

ANN	Artificial Neural Network
ATS	Automatic Text Summarization
BDT	Basque Dependency Treebank
BERT	Bidirectional Encoder Representations from Transformers
BiLSTM	Bidirectional Long Short-Term Memory
BiRNN	Bidirectional Recurrent Neural Network
BPE	Byte Pair Encoding
C4	Colossal Clean Crawled Corpus
CC	Common Crawl
CoT	Chain-of-Thought
CTG	Galician Technical Corpus
DP	Dependency Parsing
EDT	Estonian Dependency Treebank
EIEC	Basque Named Entities Corpus
ELECTRA	Efficiently Learning an Encoder that Classifies Token Replacements Accurately
ENC	Estonian National Corpus
FFN	Feedforward Neural Network
GLUE	General Language Understanding Evaluation
GMM	Gaussian Mixture Model
GPT	Generative Pre-trained Transformer
GRU	Gated Recurrent Unit
HPLT	High Performance Language Technologies
IC3	Icelandic Common Crawl Corpus
ICC	Icelandic Crawled Corpus
IGC	Icelandic Gigaword Corpus
IQR	Interquartile Range
LAS	Labeled Attachment Score
LM	Language Model
LSTM	Long Short-Term Memory
LT	Language Technology
MAD-X	Multiple Adapters for Cross-lingual Transfer
mC4	Multilingual Colossal Clean Crawled Corpus
MLM	Masked Language Modeling
mT5	Multilingual Text-to-Text Transfer Transformer
MWS	Multi-word Selection
NER	Named Entity Recognition
NLI	Natural Language Inference
NLU	Natural Language Understanding

NNC	Nepali National Corpus
NP	Noun Phrase
NSP	Next Sentence Prediction
NTP	Next Token Prediction
QA	Question Answering
OCR	Optical Character Recognition
OCSVM	One-class Support Vector Machine
OOV	Out-of-Vocabulary
PaLM	Pathways Language Model
POS	Part-of-Speech
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
RoBERTa	Robustly Optimized BERT Approach
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
RTD	Replaced Token Detection
RUQuAD	Reykjavik University Question-Answering Dataset
SLM	Statistical Language Model
T5	Text-to-Text Transfer Transformer
TC	Topic Classification
TEAMS	Training ELECTRA Augmented with Multi-word Selection
TLM	Translation Language Modeling
TPU	Tensor Processing Unit
TQ-IS	Text Quality Dataset for Icelandic
UD	Universal Dependencies
UPOS	Universal POS
VP	Verb Phrase
XLM	Cross-lingual Language Model
XNLI	Cross-lingual Natural Language Inference Corpus

Chapter 1

Introduction

The field of Natural Language Processing (NLP) has advanced at a rapid pace in recent years. Virtual assistants are now commonplace in smart devices, image generation models can create high-quality images from natural language prompts, and chatbots powered by advanced language models (LMs) can explain complex concepts, compose essays, and generate programming code. These breakthroughs have driven increased commercial investment in NLP applications, which in turn has accelerated research and funding in the field.

These advancements were enabled by the introduction of the Transformer architecture (Vaswani et al., 2017), a scalable and efficient neural network model. Originally developed for machine translation, the Transformer quickly outperformed previous approaches across a wide range of NLP tasks (Radford et al., 2018; Devlin et al., 2019). One of its defining strengths is its scalability, which allows for much larger models to be trained than was previously feasible. Larger models tend to achieve better performance, provided sufficient training data and computational resources are available. As a result, state-of-the-art models have grown exponentially in both size and training data. This development has primarily benefited high-resource languages, such as English, where extensive datasets and computing infrastructure are readily available.

In contrast, low- and medium-resource languages face considerable challenges. Web-crawled corpora, while widely used in pre-training, are often the only viable option for low-resource languages due to the scarcity of curated datasets. However, such corpora are often of dubious quality, especially for low-resource languages, frequently containing noisy, low-quality text, making effective training difficult (Kreutzer et al., 2022). An alternative approach is to leverage massively multilingual LMs, but these models often underperform for low-resource languages, as they are often underrepresented in the training data (Pyysalo et al., 2021).

To address these challenges, this thesis investigates strategies to maximize the performance of Transformer-based LMs for low- and medium-resource languages. Specifically, we explore methods for improving web-crawled training data through heuristic and machine-learning based filtering techniques, examine how multilingual augmentation can improve model performance, and analyze the impact of different subword tokenization configurations in resource-constrained settings.

1.1 Motivation

Transformer-based language models (Vaswani et al., 2017) have significantly advanced NLP, achieving state-of-the-art performance across a wide range of tasks (Radford et al., 2018; Devlin et al., 2019; Warner et al., 2024; DeepSeek-AI, 2024). These models are typically trained in two stages. First, they undergo *pre-training* on large text corpora on tasks such as learning to predict masked words or the next word in a sequence. This phase enables the model to develop general language understanding without requiring human-annotated data. Once pre-trained, the model can be *fine-tuned* on smaller, labeled datasets for more practical tasks (referred to as *downstream tasks*), such as part-of-speech (POS) tagging, question answering (QA), or automatic text summarization (ATS). Task-specific fine-tuning remains common for many Transformer-based models, although larger generative models typically undergo *instruction tuning*, which adapts them to respond to user prompts and generalize across multiple tasks rather than specializing in one. While pre-training can take anywhere from several hours to months to complete, depending on model size and available computational resources, fine-tuning typically requires far less time and computing power. Despite the costs associated with large-scale training, the effectiveness of Transformer models has made them the dominant approach in NLP.

To improve the efficiency and scalability of Transformer-based models, researchers have introduced various architectural optimizations. These include parameter-sharing techniques that reduce memory requirements (Lan et al., 2019) and improved activation functions that enhance model performance (Shazeer, 2020). As a result, Transformer models have grown exponentially in size, from 110 million parameters in GPT (2018) (Radford et al., 2018) to 671 billion in DeepSeek-V3 (2024) (DeepSeek-AI, 2024). This growth in model size has made cutting-edge performance increasingly dependent on access to substantial computational resources.

The computational costs of training state-of-the-art Transformer models have grown dramatically alongside their size. Training the full GPT-3 model (Brown et al., 2020) has been estimated to have cost $4.6 million[1], while pre-training costs for GPT-4 (OpenAI, 2024) exceeded $100 million[2]. More recent approaches have demonstrated improved efficiency, such as DeepSeek-V3, which reduced pre-training costs to $5.6 million while maintaining competitive performance (DeepSeek-AI, 2024). Despite these advances, large-scale pre-training remains prohibitively expensive for most. This creates a significant barrier for researchers working with low- and medium-resource languages, who often lack access to both extensive computing infrastructure and training data. For these languages, the key challenge becomes maximizing model performance under significant computational and data constraints. This requires careful consideration of pre-training strategies and data preparation techniques.

1.2 Research Questions

It is well established that increasing the amount of pre-training data improves downstream performance (Liu et al., 2019; Raffel et al., 2020; Muennighoff et al., 2023). Consequently, recent LMs have been trained on exponentially larger datasets, grow-

[1] https://lambdalabs.com/blog/demystifying-gpt-3/

[2] https://web.archive.org/web/20230418190335/https://www.wired.com/story/openai-ceo-sam-altman-the-age-of-giant-ai-models-is-already-over/

ing from 800 million tokens for GPT (2018) to 14.8 trillion for DeepSeek-V3 (2024). However, datasets of this scale are only available for a handful of the highest-resourced languages, while even medium-resource languages fall far short of such vast amounts of data. This creates a significant challenge in developing high-quality models for less-resourced languages.

To expand the size and diversity of pre-training corpora, many researchers rely on web-crawled data, which is abundant but presents several challenges. Online text often contains duplicate content (e.g., identical cookie notifications and privacy policies), noisy text, low-quality machine translations, and character encoding errors. Additionally, if not carefully curated, web-crawled data can introduce significant biases into pre-trained models. To mitigate these issues, various filtering techniques are applied, such as *deduplication*, heuristic rules to identify low-quality text, language classification to remove unwanted languages, and filtering out text containing offensive content.

Filtering web-crawled corpora has been shown to improve the downstream performance of pre-trained LMs (Brown et al., 2020; Wenzek et al., 2020). However, direct comparisons between models pre-trained on filtered and unfiltered corpora remain rare and are largely restricted to the English language. Furthermore, these experiments rarely provide a fine-grained analysis of the impact of individual filtering techniques or heuristic rules. This leads us to our first research question:

RQ1: How do different text filtering techniques impact the downstream performance of LMs pre-trained on web-crawled corpora for low- and medium-resource languages?

Increasing the size of monolingual corpora for low-resource languages with more text in the same language is not always a viable option. An alternative approach is to use multilingual models, which are pre-trained on corpora containing text from multiple languages. One key advantage of these models is that a single multilingual model can replace numerous monolingual models, including for languages where no monolingual model exists. Another potential benefit is cross-lingual transfer, where knowledge learned from one language benefits another. While multilingual Transformer models have shown moderate success, they often struggle to outperform monolingual models for low-resource languages (Pyysalo et al., 2021; Dađason and Loftsson, 2022; Garcia, 2024). One likely reason is that these models are typically trained on an extremely large number of languages, often over a hundred (Conneau et al., 2020; Xue et al., 2021; Anil et al., 2023). In such large-scale multilingual settings, low-resource languages are inevitably underrepresented in the training corpora, which can negatively impact their downstream performance (Conneau et al., 2020).

A notable example of a multilingual model is mBERT[3], released by the authors of BERT. It was trained on Wikipedia articles in 104 of the largest languages on the online encyclopedia. While mBERT performs reasonably well for many languages, Pyysalo et al. (2021) demonstrate that in a majority of cases, a monolingual BERT model trained on Wikipedia articles achieves comparable or better downstream performance. Additionally, Wu and Dredze (2020) show that for the 30% of its lowest-resource languages, mBERT is outperformed by models based on older neural network architectures, such as bidirectional long short-term memory (BiLSTM) networks. They also find that bilingual models trained on closely related languages can sometimes outperform both mBERT and monolingual models, suggesting that careful selection of training languages may be more important than maximizing language coverage.

[3] `https://github.com/google-research/bert/blob/master/multilingual.md`

As mentioned earlier, while large multilingual models achieve reasonable downstream performance on average, they often underperform for low-resource languages (Pyysalo et al., 2021). A more effective approach may be to design multilingual models that specifically optimize the performance of a target low-resource language rather than aiming for broad coverage across many unrelated languages. This could involve carefully selecting languages for inclusion in the pre-training corpus while ensuring that the target language remains well represented. One promising criterion for selection is language similarity, which has been hypothesized to be a key factor in cross-lingual transfer (Wu and Dredze, 2020; Snæbjarnarson et al., 2023). However, the relative value of different types of additional training data, whether from related languages, unrelated languages, or even structured data, remains poorly understood. This brings us to our next research question:

RQ2: How does linguistic similarity influence the effectiveness of cross-lingual transfer in bilingual models?

Out-of-vocabulary (OOV) words are a major source of errors in many NLP models. Transformer-based models address this issue using subword tokenizers, which break unknown words into sequences of known subwords. These tokenizers operate with a fixed-size vocabulary, typically learned from the pre-training corpus. Models such as BERT and ELECTRA (Clark et al., 2020) use characters as the smallest subword units, while others, such as RoBERTa (Liu et al., 2019) and GPT-3, use Unicode bytes instead, eliminating OOV words altogether. While subword tokenization is commonly based on the byte-pair encoding (BPE) algorithm (Sennrich et al., 2016), alternative approaches, such as the unigram algorithm (Kudo, 2018), have shown some promise in certain settings (Zhang et al., 2020). The choice of tokenization strategy may be particularly consequential for low-resource languages, which often feature complex morphology and limited training data for learning optimal subword splits.

Research on the impact of vocabulary size and tokenization algorithms has produced conflicting results, particularly for low-resource languages. For a Turkish BERT model pre-trained on a 44B token corpus, Schweter (2020) found no significant difference in downstream performance between a 32k and a 128k vocabulary. In contrast, Wu and Dredze (2020) observed that for some low-resource languages, a smaller vocabulary size yielded better results. These contradictory findings suggest that the relationship between subword tokenizer configuration and model performance may depend on language-specific factors such as morphological complexity and the size of available training data. This leads to our third research question:

RQ3: How do different subword tokenization algorithms and vocabulary sizes impact downstream performance for low- and medium-resource languages?

1.3 Thesis Contribution

In this section, we outline our main contributions. We begin by presenting the datasets, corpora, and tools developed and released during our research, along with the LMs that we pre-trained and published:

- We released IceSum, a dataset of 1,000 Icelandic news articles annotated with extractive summaries. This is the first Icelandic corpus for ATS featuring human-annotated summaries. We included this dataset in the NLP benchmark used

to evaluate Icelandic LMs in our experiments. IceSum is distributed on the CLARIN-IS repository with an open license[4] (Daðason et al., 2021).

- We trained multiple Transformer-based language models covering 13 languages: Icelandic, Norwegian, Danish, Swedish, Finnish, Estonian, Basque, Spanish, Galician, Portuguese, Nepali, Hindi, and Tajik. These include monolingual, bilingual, and multilingual variants using the ELECTRA, ConvBERT (Jiang et al., 2020), and TEAMS (Shen et al., 2021) architectures in both Small and Base configurations. All models were released under an open license at the Hugging Face model repository.[5]

- We published IceEval, a tool for automatically fine-tuning and evaluating local Transformer-based language models on a benchmark of four Icelandic NLP tasks: POS tagging, named entity recognition (NER), dependency parsing (DP), and ATS. We released it with an open license at the CLARIN-IS repository[6] We adapted this tool to evaluate LMs in multiple different languages for this thesis (Daðason and Loftsson, 2022).

- We created and published TQ-IS, a text quality dataset for Icelandic consisting of 2,000 web-crawled documents manually annotated with fine-grained text quality labels at the span level, such as "incoherent text", "foreign text", and "nonlinguistic text". Additionally, each document was assigned an overall quality label (low or high). To the best of our knowledge, this is the first publicly available text quality dataset in any language consisting of full documents manually annotated with text quality labels, suitable for training and evaluating document-level text quality classifiers. TQ-IS is available on GitHub with an open license[7] (Daðason and Loftsson, 2024).

- We released the Icelandic Crawled Corpus (ICC), comprising 930M tokens of Icelandic text collected from websites using ad-hoc crawlers. We targeted domains that had no or minimal representation in existing datasets. It consists primarily of online forum posts, news articles, blog entries, and adjudications from government agencies. Documents in the TQ-IS dataset were, in part, sampled from the ICC. The corpus is available on the Hugging Face dataset repository[8] (Daðason and Loftsson, 2024).

Our contributions to data-efficient pre-training techniques, specifically in text quality filtering for noisy web-crawled corpora and multilingual augmentation strategies for low- and medium-resource languages, can be summarized as follows:

- We evaluated 12 commonly used heuristic rules for text quality classification in large-scale, multilingual web-crawled corpora. Additionally, based on insights from the TQ-IS dataset, we introduced a novel rule leveraging subword tokenizer statistics to detect low-quality text. Our evaluation on TQ-IS demonstrated that, with optimized threshold values, heuristic rules are highly effective in filtering out

[4]IceSum: `http://hdl.handle.net/20.500.12537/285`
[5]Pre-trained LMs: `https://huggingface.co/jonfd`
[6]IceEval: `http://hdl.handle.net/20.500.12537/297`
[7]TQ-IS: `https://github.com/jonfd/tq-is`
[8]ICC: `https://huggingface.co/datasets/jonfd/ICC`

low-quality documents. Notably, we found that a combination of just three rules achieved near-optimal F_1 scores for text quality classification. This suggests that complex rule-based filtering pipelines could be simplified without compromising quality.

- We implemented and trained several previously described text quality classifiers and evaluated their performance on TQ-IS. Our findings indicate that these classifiers either underperform compared to heuristic rules or require manually annotated text quality datasets for training and fine-tuning, which are generally not available for most languages.

- We proposed a novel text quality classification approach using unsupervised clustering and outlier detection algorithms. These methods require no manually annotated data and can be tuned efficiently, even without expertise in the target language. Our best-performing unsupervised classifier achieved F_1 scores comparable to the rule-based approach while being highly computationally efficient.

- We applied the best-performing unsupervised text quality classifier to filter noisy web-crawled corpora for six diverse low- to medium-resource languages. We then pre-trained LMs on both filtered and unfiltered corpora and evaluated their performance on a benchmark of NLP tasks. Our results show that despite the filtered corpora containing 42–81% less data (depending on the language), the models trained on them performed similarly or better across almost all tasks.

- We explored multilingual augmentation strategies by supplementing an Icelandic corpus with text from three different sources: Norwegian (a closely related language), Finnish (an unrelated language), and Python code. We then pre-trained LMs on the augmented corpora and found that the Norwegian-augmented model outperformed other bilingual models and performed similarly to a monolingual model. This finding supports the hypothesis that language similarity facilitates cross-lingual transfer.

- We augmented monolingual corpora for five low- to medium-resource languages with text from a donor language and pre-trained bilingual models. Comparing their performance against monolingual models on a benchmark of NLP tasks, we observed that bilingual models generally achieved similar or slightly lower results. However, models pre-trained with more closely related languages performed slightly better than those trained with unrelated languages.

- We evaluated three subword tokenization algorithms at different vocabulary sizes on Icelandic and analyzed their impact on downstream performance across multiple NLP tasks. Our findings showed that vocabulary size had a greater influence than the choice of algorithm, with the impact varying by task. We then confirmed these trends in a cross-lingual evaluation across five other morphologically rich, low- to medium-resource languages.

- We compared character-level and byte-level tokenization in a benchmark of Icelandic NLP tasks and found no statistically significant difference in downstream performance, indicating that byte-level tokenization provides little advantage for high-quality monolingual corpora.

- Across our experiments, we pre-trained over 44 language models of varying sizes, evaluated them on 20 distinct NLP tasks, and conducted more than 2,200 fine-tunings runs to obtain our final results.

1.4 Structure of the Thesis

Chapter 2 introduces key concepts and terminology used throughout this thesis. We provide a brief overview of subword tokenization, explaining its necessity and implementation of common algorithms. We then outline the characteristics of Transformer-based LMs, explaining how they work in high-level terms, and contrast them with recurrent neural networks (RNN) and statistical models. Finally, we detail the hyper-parameters used for pre-training and fine-tuning our LMs.

In Chapter 3, we describe the six diverse languages selected for our evaluations and explain why they were chosen. We also provide an overview of the corpora, datasets, tools, and libraries used in our experiments.

Chapter 4 explores various text quality filtering techniques, including heuristic rules and supervised, weakly supervised, and unsupervised classifiers. We evaluate these techniques on TQ-IS and select the most suitable method for filtering noisy web-crawled corpora across all six languages. Subsequently, we pre-train LMs on both filtered and unfiltered corpora and evaluate their downstream performance on our NLP benchmark tasks.

Chapter 5 presents our experiments with multilingual models. We augment low- and medium-resource corpora with text from related and unrelated languages, as well as programming code. After pre-training LMs on these augmented corpora, we evaluate their performance on our NLP benchmark and analyze the relative impact of different data sources.

In Chapter 6, we evaluate various character-level and byte-level tokenization algorithms with different configurations, such as vocabulary size, and compare the impact that they have on downstream performance for a diverse set of languages.

Finally, Chapter 7, summarizes our findings, presents our conclusions, and discusses potential directions for future research.

1.5 Publications

The following papers, published in peer-reviewed conference proceedings, describe or are relevant to some of the findings presented in this thesis:

1. Jón Daðason, Hrafn Loftsson, Salome Sigurðardóttir, and Þorsteinn Björnsson. 2021. IceSum: An Icelandic Text Summarization Corpus. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 9–14, Online. Association for Computational Linguistics.

2. Jón Friðrik Daðason and Hrafn Loftsson. 2022. Pre-training and Evaluating Transformer-based Language Models for Icelandic. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 7386–7391, Marseille, France. European Language Resources Association.

3. Jón Daðason and Hrafn Loftsson. 2024. Text Filtering Classifiers for Medium-Resource Languages. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 15789–15801, Torino, Italia. European Language Resources Association and International Committee on Computational Linguistics.

4. Jón Daðason and Hrafn Loftsson. 2024. Unsupervised Outlier Detection for Language-Independent Text Quality Filtering. In *Proceedings of the 3rd Annual Meeting of the Special Interest Group on Under-resourced Languages @ LREC-COLING 2024*, pages 383–393, Torino, Italia. European Language Resources Association and International Committee on Computational Linguistics.

Chapter 2

Natural Language Processing

In this chapter, we describe several fundamental concepts and techniques in Natural Language Processing (NLP) that are relevant to this thesis. We begin by discussing subword tokenization methods, followed by an overview of different language modeling approaches. We describe statistical models and neural architectures, including recurrent neural networks (RNNs) and Transformers, comparing bidirectional and unidirectional approaches. We conclude with a methodology section, detailing the settings we used for pre-training, fine-tuning, and evaluating our language models (LMs).

2.1 Subword Tokenization

Tokenization is the process of breaking text into smaller units called *tokens*, which can include words, punctuation marks, and symbols. In NLP, this step is often performed alongside *sentence segmentation*, which involves dividing text into sentences. This dual process is commonly used to prepare datasets for tasks such as part-of-speech (POS) tagging and named entity recognition (NER), where each token is annotated with a corresponding tag. Such tasks are collectively referred to as token classification tasks.

These token classification tasks typically employ *supervised learning*, where models are trained on human-annotated datasets to predict appropriate tags for new text. While these models generally perform well on frequently occurring tokens, they face significant challenges when encountering tokens that rarely appear in the training data, or out-of-vocabulary (OOV) tokens that have never been seen. Due to this limitation, models tend to perform better when trained on diverse datasets and using techniques to handle rare and OOV tokens effectively.

Early NLP models primarily operated at the word level, associating entire tokens, such as "photochemical", with specific tags such as "adjective" (Bird et al., 2009). This approach was effective for tokens present in the training data but struggled with OOV tokens. When encountering such tokens, these models relied on heuristic strategies, such as analyzing suffixes (e.g., "-ical" indicates an adjective). These methods often resulted in lower accuracy due to their limited ability to generalize.

To address the limitations of word-level models in handling OOV tokens, Sennrich et al. (2016) introduced an approach to subword tokenization based on the Byte Pair Encoding (BPE) algorithm (Gage, 1994). Subword tokenization breaks previously unseen words into smaller and more manageable units, enabling models to process them more effectively. The BPE algorithm begins with a minimal vocabulary derived from a given corpus, such as individual characters. It then iteratively merges the most

frequent pairs of units, forming larger subwords, until a predefined vocabulary size is reached. This approach ensures that OOV tokens can be broken down into known subwords. For instance, if the word "photochemical" is not in the vocabulary, it might be decomposed into the subwords "photo" and "chemical", allowing the model to make predictions using familiar components. Subword tokenization has become a standard technique in modern NLP due to its ability to balance vocabulary size with coverage, enabling neural models to generalize better across diverse data.

However, subword tokenization introduces certain trade-offs. Modern Transformer-based models have a limited context size, which determines the maximum number of tokens they can process at once. Since subword tokenization often splits words into multiple subwords, it can result in longer token sequences, effectively reducing the usable context size for a given input. For example, a sentence containing rare or complex words might require more subwords to represent, leaving less room for additional context. Larger vocabularies can help mitigate this issue by reducing the average number of subwords per word, but they come with increased computational cost and memory requirements.

Beyond BPE, other subword tokenization algorithms have been developed. *WordPiece* (Wu et al., 2016) builds its vocabulary by maximizing the likelihood of the training data, while the *Unigram* algorithm (Kudo, 2018) selects subwords based on a probabilistic model. Additionally, some tokenizers, such as the one used by the GPT-2 model (Radford et al., 2019), use individual bytes rather than characters as the smallest possible units. By operating at the byte level, these tokenizers completely eliminate OOV tokens (including characters that did not occur in the training data) and provide a simple yet effective solution for handling diverse scripts and character sets.

These algorithms and techniques are described and evaluated in detail in Chapter 6.

2.2 Language Models

An LM is a probabilistic model that predicts words or fragments of text based on patterns and probabilities learned from text corpora. For example, given the text sequence "Nice to", an LM might predict that "meet" is the most likely next word, followed by "you." This process is characteristic of autoregressive LMs, which generate text by predicting one token at a time, using each newly generated token as additional context for subsequent predictions.

This type of LM is also described as being unidirectional because it processes text in a single direction, typically using only the preceding tokens as context (i.e., left-to-right or right-to-left). In contrast, bidirectional LMs utilize context from both directions, considering tokens both before and after a given position to make more informed predictions. Each approach has its own advantages and limitations, which are discussed later in this section.

While text generation is one of the most intuitive applications of LMs, their ability to model contextual relationships between tokens makes them valuable for a wide range of NLP tasks. For example, LMs can be trained on text corpora annotated with linguistic information, such as POS tags. By learning patterns and probabilities associated with token sequences and their corresponding linguistic features, these models can then predict POS tags for previously unseen, unannotated text.

2.2.1 Statistical Models

Statistical language models (SLMs) assign probabilities to sequences of tokens based on their frequency distributions in a training corpus. Among the most widely used SLMs are *n-gram* models, which estimate probabilities for sequences of n consecutive tokens, such as words or characters.

An n-gram represents any sequence of n adjacent tokens. The value of n determines the model's order: a bigram ($n = 2$) model predicts tokens based on pairs of consecutive tokens, while a trigram ($n = 3$) model uses sequences of three tokens, and so on.

Intiutively, higher-order n-gram models should perform better since they consider more context when making predictions. However, as n increases, the number of possible n-gram combinations grows exponentially. This leads to a data sparsity problem, where many valid n-gram sequences do not appear in the training corpus, even when it is large. As a result, the model cannot reliably estimate their probabilities, effectively limiting the maximum practical order of n-gram models.

To mitigate this sparsity issue, n-gram models typically employ smoothing techniques, which adjust probability estimates to account for unseen sequences. One widely used method is Kneser-Ney smoothing (Kneser and Ney, 1995), which uses lower-order n-gram models to estimate probabilities for sequences that were not observed in the training corpus. For example, when encountering an unseen trigram sequence, the model might back off to bigram or unigram probabilities to estimate its likelihood.

In addition to smoothing, systems using n-gram models can employ additional strategies to handle unseen tokens and sequences. For example, an SLM-based POS tagger might analyze the suffixes of unseen words (e.g., "-ing" or "-ly") to infer their likely tag. Although smoothing and other techniques help mitigate the sparsity problem, they cannot fully overcome the fundamental limitations of n-gram models, such as their limited ability to capture long-range dependencies in language.

2.2.2 Recurrent Neural Networks (RNNs)

Artificial neural networks (ANNs) have proven to be highly effective for tasks such as NLP and computer vision. These networks are composed of interconnected artificial neurons arranged in layers. An ANN typically consists of an input layer, which receives numerical values as input; one or more hidden layers, where intermediate computations occur; and an output layer that generates predictions for a particular task. The multiple hidden layers allow the network to learn hierarchical features, enabling it to capture complex patterns in the data.

Passing input values through the network involves several mathematical operations, most notably matrix multiplication and addition. Each neuron receives inputs from neurons in the previous layer, with each connection having a certain weight. To compute the value for a neuron, its inputs are multiplied by their respective weights, and a bias term associated with the neuron is added to the sum. This result is passed through an activation function, such as the Rectified Linear Unit (ReLU) (Agarap, 2019), which introduces non-linearity and enables the network to learn complex patterns. The outputs of these calculations become inputs for the next layer, and this continues until the final predictions are generated by the output layer. This process is known as a *forward pass*.

Weights and biases are trainable parameters of the model, meaning that they are typically initialized with random values and gradually adjusted during training to

improve the accuracy of the model's predictions. During training, after a forward pass with one or more training examples, the model's predictions are compared against the correct labels using a *loss function*, which quantifies the error. For example, in a classification task like predicting whether a movie review is positive or negative, the model might output probabilities for each class. The loss function measures how far the predicted probabilities deviate from the correct answer.

After calculating the loss, an algorithm such as *backpropagation* (Rumelhart et al., 1986) is used to estimate how much each weight and bias contributed to the error. This involves computing gradients, which indicate the direction and magnitude of the change needed to reduce the error. These gradients are used by an optimization algorithm, such as stochastic gradient descent, to update the weights and biases in a way that reduces the error. This process repeats over many training examples, allowing the model to gradually improve its predictions.

Since ANNs accept only numerical values as input, plain text sequences must first be converted into a numerical format. Typically, each word is represented as a *word embedding* (also known as a word vector), a fixed-size vector of real numbers. These embeddings are stored in a separate embedding layer, and a lookup table is used to replace words with their vector representations. Like weights and biases, word embeddings are trainable parameters that are adjusted during training. As the training progresses, the embeddings gradually capture semantic and syntactic information about the words. Pre-trained embeddings, such as Word2Vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014), can also be used to initialize the embedding layer. These provide a strong starting point for training since they already encode useful semantic and syntactic relationships. While such pre-trained embeddings can be further fine-tuned, they are often frozen (not changed) during the training process to preserve this learned information.

RNNs are a subcategory of ANNs designed specifically for sequential data, such as text, time series, or speech. While traditional non-recurrent models process inputs independently, RNNs maintain an internal memory (referred to as the *hidden state*) that captures information about previously processed inputs. As they process a sequence one element at a time, RNNs compute a hidden state at each step, which reflects the current input and contextual information from previous steps. For tasks like sentiment analysis, an RNN generates predictions based on the final hidden state of the input sequence. For token classification tasks, such as POS tagging, it produces predictions for individual words based on their corresponding hidden states. However, standard RNNs can struggle with long sequences due to vanishing gradients, a limitation addressed by more recent variants such as Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Units (GRUs) (Cho et al., 2014).

The embedding layer contains static embeddings, meaning that each word is represented by the same vector regardless of its context. For example, the word "rock" has the same embedding whether it refers to a mineral formation or the musical genre. However, RNNs incorporate contextual information by using previous words to refine hidden states that are derived from these embeddings. This contextual representation can be further enhanced by using bidirectional RNNs (BiRNNs), which process the input sequence in both forward and backward directions. By concatenating the hidden states from the forward and backward passes, BiRNNs capture contextual information from both preceding and succeeding words.

2.2.3 The Transformer Architecture

The Transformer (Vaswani et al., 2017) is a neural network architecture originally developed for sequence-to-sequence tasks such as machine translation. It uses an encoder-decoder structure, where the encoder processes an input sequence (e.g., text in one language) to produce a contextual representation, which the decoder then uses to generate an output sequence (e.g., the same text in another language).

A key feature of the Transformer is the *self-attention* mechanism, which models relationships between token pairs in an input sequence, allowing the model to capture contextual information across long distances. It also enables the model to process input sequences in parallel, rather than sequentially, as is the case with traditional RNNs, significantly improving efficiency and scalability.

Self-attention computes how much each token should "attend to" other tokens in the sequence, including itself. For example, it might find a strong relationship between a pronoun and its antecedent. These attention patterns are learned through three learnable projections for each token: queries, keys, and values. The attention scores are computed by comparing each token's query vector with the key vectors of the other tokens, and these scores are used to create a weighted combination of the value vectors. The word embeddings are enriched with contextual information from the attention outputs.

The original Transformer architecture consists of multiple encoder and decoder layers, each employing multi-head self-attention, where multiple attention "heads" apply the self-attention mechanism to the input in parallel. During training, each head learns to focus on different linguistic or task-specific characteristics of the input. For example, one head might learn to attend to direct objects of verbs, while another might attend to coreferent mentions (Clark et al., 2019). The outputs of these heads are combined to produce a more comprehensive representation of the input.

Although the self-attention mechanism offers significant advantages, the number of operations required grows quadratically with the length of the input sequence (Vaswani et al., 2017). Doubling the number of tokens in a sequence results in four times as many computations being performed. This quadratic complexity poses challenges for models handling very long inputs. Increasing the maximum sequence length of a model becomes prohibitively expensive beyond a certain point. Various strategies, such as sparse attention mechanisms (Child et al., 2019), can be used to limit the cost of the attention mechanism, although they often involve trade-offs with regard to the overall effectiveness of the model.

2.2.3.1 Encoder-Only/Bidirectional Models

Bidirectional Transformer-based LMs depart from the encoder-decoder structure of the original Transformer architecture, instead relying solely on encoder layers (Devlin et al., 2019). These models utilize bidirectional self-attention, allowing each token to incorporate contextual information from the entire input sequence.

In the Transformer's encoder layers, the outputs of multiple attention heads are concatenated and combined with the input embeddings using residual connections, enriching them with contextual information. Layer normalization is then applied to the embeddings, maintaining them in a standardized range as they pass through the layers, stabilizing training and speeding up convergence. Then, a feedforward neural

network (FFN) refines the embeddings further, with another residual connection and layer normalization step before the embeddings are passed to the next encoder layer.

After passing through all encoder layers, the embeddings can be used by a task-specific head to generate predictions. For example, in text classification tasks such as sentiment analysis, the head computes the probability of each label (e.g., positive or negative sentiment) being correct for the input based on the embeddings.

Unlike the original Transformer model, which was trained on parallel corpora for machine translation, bidirectional models are typically first pre-trained on large, unannotated corpora using *self-supervised learning* (Devlin et al., 2019; Liu et al., 2019; Clark et al., 2020). This type of learning involves tasks where the labels are derived directly from the unannotated data itself. Pre-training is typically the most time-consuming and computationally expensive part of training the model, but allows it to learn generalizable language representations.

The Masked Language Modeling (MLM) pre-training task, introduced by the BERT LM (Devlin et al., 2019), is widely used for pre-training bidirectional models. For this task, a portion of the input tokens is masked, and the model attempts to predict the original forms of the masked tokens, selecting from the full vocabulary.

Once pre-trained, the model can be fine-tuned on annotated datasets for downstream tasks such as POS tagging, NER, and question answering (QA). Fine-tuning generally requires much less time and computational resources than pre-training, as the model has already learned a strong language representation. The results obtained on these downstream tasks are referred to as the model's downstream performance.

In the following list, we describe several bidirectional, encoder-only Transformer-based LMs:

BERT BERT (Devlin et al., 2019) is the first bidirectional, encoder-only LM based on the Transformer architecture. It was pre-trained on two tasks: MLM and Next Sentence Prediction (NSP). In the NSP task, the model predicts whether the second sequence in a pair of text sequences immediately follows the first. BERT combines the losses from MLM and NSP during pre-training to learn contextual and sequential relationships. It was pre-trained on 3.2 billion words from English-language encyclopedic articles and self-published books, with two released variants: BERT-Base (110M parameters) and BERT-Large (340M parameters). A multilingual version of BERT-Base (mBERT) was also published, pre-trained on Wikipedia articles in 104 languages.

RoBERTa RoBERTa (Liu et al., 2019) builds on BERT by refining the pre-training process. It removes the NSP task, as it was not found to have a positive impact on downstream performance. It also employs dynamic masking, where input sequences are randomly masked during each training epoch, rather than during a pre-processing step. Additionally, RoBERTa increases batch sizes, processes longer input sequences, and is pre-trained on 160 GB of English-language text from diverse genres, ten times the size of BERT's pre-training corpus. Two variants of the RoBERTa model were released: RoBERTa-Base (125M parameters) and RoBERTa-Large (355M parameters).

ELECTRA ELECTRA (Clark et al., 2020) introduces the Replaced Token Detection (RTD) task, where the model must identify tokens in an input sequence that have been replaced with less plausible alternatives. The model uses a generator-discriminator architecture composed of two LMs that are pre-trained concurrently.

The generator, which is the smaller of the two models, is pre-trained on the MLM task. Tokens in the input sequence are masked and replaced with the generator's predictions, though its accuracy is limited by its size. The larger discriminator then predicts which tokens are original and which have been replaced. RTD differs from MLM in that it provides a learning signal for every token in the input sequence, rather than only the 15% of tokens that are typically masked. This characteristic has been hypothesized to improve data efficiency (Wu and Dredze, 2020), but direct empirical comparisons remain limited. ELECTRA was pre-trained on the same corpus as BERT, except for its largest variant, which was pre-trained on 33B tokens. Three variants of the ELECTRA model were released: ELECTRA-Small (14M parameters), ELECTRA-Base (110M parameters), and ELECTRA-Large (335M parameters).

ConvBERT ConvBERT (Jiang et al., 2020) modifies the ELECTRA architecture by introducing an efficient hybrid attention mechanism. It reduces the computational cost of self-attention by projecting input embeddings into a lower-dimensional space, decreasing the number of attention heads while maintaining high-quality representations. Additionally, ConvBERT employs a span-based dynamic convolution mechanism that replaces some attention heads to more efficiently model local dependencies. In this context, a convolution refers to a mathematical operation that applies a sliding filter (or kernel) over a sequence to extract local patterns, such as relationships between neighboring tokens. ConvBERT dynamically generates these kernels based on the context of each token. The outputs of the dynamic convolution mechanism are combined with reduced self-attention in a mixed attention block, effectively balancing global and local context modeling. These modifications improve the model's efficiency and downstream performance. The model was pre-trained on 38 GB of web-crawled English-language text. Three variants of ConvBERT were released: ConvBERT-Small (14M parameters), ConvBERT-Medium-Small (17M parameters), and ConvBERT-Base (106M parameters).

TEAMS TEAMS (Shen et al., 2021) is another modification of the ELECTRA model, which introduces a second pre-training task, Multi-Word Selection (MWS). In addition to identifying replaced tokens, the model predicts the original form of each replaced token from a set of five candidates. The authors find that TEAMS models outperform ELECTRA consistently across a benchmark of NLP tasks. The TEAMS models were pre-trained on the same corpora as the ELECTRA models. The authors release the TEAMS models in three sizes: TEAMS-Small (14M parameters), TEAMS-Base (110M parameters), and TEAMS-Large (335M parameters).

2.2.3.2 Decoder-Only/Unidirectional Models

Generative Transformer-based LMs are structurally similar to bidirectional models but rely on decoder layers instead of encoder layers. The most significant difference between them is that the decoder layer implements causal self-attention. Unlike bidirectional models, which compute relationships between all tokens in an input sequence, causal self-attention restricts each token to attend only to itself and preceding tokens. This unidirectional approach aligns with the requirements of text generation tasks, ensuring that predictions depend only on prior context.

These models are typically pre-trained using the Next Token Prediction (NTP) task, where the objective is to predict the next token in a sequence based on the

preceding tokens. When trained on this task, the model learns how to generate coherent text, making it well-suited for applications such as language modeling, creative writing, and chatbot development.

Prominent examples of decoder-only models include the GPT and Llama families:

GPT The original Generative Pre-trained Transformer (GPT) model (Radford et al., 2018), with 117M parameters, was pre-trained on 800M tokens of self-published books in English. It was followed by the release of GPT-2 (Radford et al., 2019), which expanded both model size and training data, with between 175M to 1.5B parameters pre-trained on 40 GB of web-crawled text. GPT-3 (Brown et al., 2020) scaled to between 125M to 175B parameters and was pre-trained on 300B tokens, mostly consisting of web-crawled text. Information on the pre-training data or model specifications of subsequent versions of GPT has not been published. GPT-3.5, which was used to power the ChatGPT chatbot, was made commercially available in late 2022, followed by GPT-4 (OpenAI, 2024) in 2023, and GPT-4o, a multimodal version of GPT-4, in 2024.

Llama The first model in the Llama (Large Language Model Meta AI) series (Touvron et al., 2023a), ranging from 7B to 65B parameters in size, was pre-trained on 1.2T tokens, predominantly consisting of web-crawled English-language text, but also GitHub repositories and Wikipedia articles in 20 languages. Llama 2 (Touvron et al., 2023b), with 7B to 70B parameters, was pre-trained on 1.8T tokens, with approximately 90% of the data consisting of English-language text. The following version of the model, Llama 3 (Grattafiori et al., 2024), was pre-trained on 15T tokens of multilingual text, with model sizes ranging from 8B to 405B parameters in size.

2.2.3.3 Bidirectional vs. Unidirectional Models

Recently, there has been a marked shift in research focus from bidirectional encoder-only models to larger generative, unidirectional decoder-only models. While these generative models have demonstrated impressive capabilities across many tasks, their evaluation has primarily focused on generative applications, and they are often only compared against other generative models. This shift in focus raises the question of how these increasingly capable generative models compare against bidirectional models in classification tasks where the latter have traditionally excelled.

There is a growing literature comparing these two categories of models. However, fundamental differences in scale and architecture preclude an evaluation under identical settings. Rather, these experiments compare best-of-class models against each other, assessing real-world performance and efficiency.

Yu et al. (2023) demonstrate that bidirectional models not only remain competitive but frequently outperform much larger generative models on English-language classification tasks such as NER, political ideology prediction, and misinformation detection. For example, RoBERTa-Large achieves an F_1 score of 94.3% on NER, while Llama 2-70B obtains 82.5%. Compared to Llama 2-70B, a RoBERTa model was fine-tuned in 5.4% of the time (measured in seconds), while consuming only 3.7% of the energy (measured in kWh) at 2.6% of the cost. During inference, the cost and energy consumption of the RoBERTa model was just 1.7% of the Llama 2-70B model. Compared with the GPT-3.5 Turbo and GPT-4 models, the RoBERTa model made predictions at only 6.3% and 0.002% of their respective costs.

Zhong et al. (2023) evaluated bidirectional models, including RoBERTa-Large, against ChatGPT (using GPT-3.5) on GLUE (Wang et al., 2018), an English-language natural language understanding (NLU) benchmark, which encompasses tasks such as sentiment analysis, linguistic acceptability, and natural language inference. RoBERTa-Large obtained similar or better performance on most tasks, achieving an average score of 87.8, compared to 78.7 for ChatGPT. With 5-shot chain-of-thought (CoT) prompting (Wei et al., 2024), where the prompt includes five examples of similar problems solved using step-by-step reasoning, the average score obtained by ChatGPT improved to 86.2. However, CoT prompting significantly increases the number of tokens that must be processed and generated, resulting in greater computational inefficiency. The authors do not compare inference times, cost, or energy usage between models.

More recently, Bucher and Martini (2024) compared the downstream performance of bidirectional models, including RoBERTa, and ELECTRA, against generative models such as GPT-3.5 Turbo, GPT-4, and Claude-Opus. Their results reveal substantial and consistent advantages for bidirectional models in English and German-language classification tasks. For example, RoBERTa-Large obtained 92% accuracy in sentiment analysis of news articles, compared to 87% for GPT-4. On stance classification of political tweets, RoBERTa-Large achieved 94% accuracy, significantly outperforming GPT-4, which obtained 58%. Similarly, for emotion classification, RoBERTa-Large obtained 88% accuracy, while GPT-4 only achieved 20%. Finally, in multi-class stance classification of political positions, RoBERTa-Large obtained 88% accuracy, while GPT-4 achieved 38%.

The empirical evidence across these studies conclusively demonstrates that fine-tuned bidirectional models not only match or exceed the performance of much larger generative models on classification tasks, but do so with dramatically improved computational and energy efficiency. Although generative models may continue to advance rapidly, many of their architectural improvements have already been integrated into bidirectional models, such as ModernBERT (Warner et al., 2024). As such, it is still quite possible that bidirectional models will maintain their advantage for the foreseeable future. Their combination of superior performance, lower computational requirements, and reduced costs makes them a compelling choice for classification tasks.

Furthermore, while cutting-edge generative models can achieve competitive results on classification tasks in high-resource languages, their performance drops sharply for low-resource languages, which are poorly represented in their pre-training corpora. Empirical studies show that in these settings, they often underperform compared to other approaches, including fine-tuned bidirectional models (Robinson et al., 2023; Ahuja et al., 2023; Abdelali et al., 2024). This suggests that bidirectional models, which are significantly more data- and parameter-efficient, may be even better suited under resource-constrained settings.

2.3 Methodology

In this section, we describe how we pre-trained, fine-tuned, and evaluated LMs for our experiments.

2.3.1 Pre-Processing

For each model, we trained a WordPiece tokenizer with a vocabulary size of 32,000 on the pre-training corpus. This choice aligns with common practice in BERT-based models, which often employ vocabulary sizes ranging from 30,000 to 50,000 subwords, as demonstrated in BERTje for Dutch (30,073), CamemBERT for French (32,000), BERTeus for Basque (50,099), and FinBERT for Finnish (50,105) (de Vries et al., 2019; Martin et al., 2020; Agerri et al., 2020; Virtanen et al., 2019). A more detailed exploration of vocabulary sizes is provided in Chapter 6.

To prevent rare Unicode characters from taking up a significant portion of the vocabulary, we limited the number of single-character subwords to 1,000. We also excluded tokens that occurred only once in the training corpus to reduce noise from foreign words, arbitrary numbers, and other infrequent elements. During pre-processing, each document was normalized by removing all Unicode control characters while preserving original casing and accents. All tokenizers were trained using the *tokenizers* library for Python[1] (Moi and Patry, 2023), unless otherwise specified.

To generate pre-training examples, we adopted the *full-sentences* packing strategy used by the RoBERTa model (Liu et al., 2019). This method involves continually sampling text from the pre-training corpus to construct examples. Documents may cross example boundaries, and each example can contain text from multiple documents. This approach minimizes wasted text in the pre-training corpus and eliminates the need for padding within examples.

2.3.2 Pre-Training Settings

We pre-trained TEAMS-Small and TEAMS-Base models using TPU v4-8 accelerators and the TensorFlow Model Garden repository[2] (Yu et al., 2020) with the same settings as described by Shen et al. (2021). TEAMS-Small models, consisting of 14M parameters, were pre-trained for 500,000 steps, taking approximately 18 hours to complete. TEAMS-Base models, with 110M parameters, were trained for 1M steps, requiring around 118 hours. Both models use a maximum sequence length of 512 and were trained with a batch size of 256. Default pre-training parameters are used for all experiments unless otherwise specified.

2.3.3 Fine-Tuning Settings

As the experiments in this thesis involved several thousand fine-tuning runs across a large number of pre-trained models, languages, and tasks, extensive hyperparameter optimization was not feasible. Instead, we used a fixed set of hyperparameters for each task across all models to ensure fair and consistent comparisons. While this approach may not yield optimal performance for every individual model, it allows for a controlled evaluation across different settings.

For POS tagging, NER, DP, and ATS, we adopted the hyperparameters from Daðason and Loftsson (2022), which evaluated these tasks on Icelandic using several different bidirectional models of a comparable size. These settings performed well across all languages, so further optimization was unnecessary, unless otherwise noted. For QA and topic classification (TC), which were not covered in Daðason and Loftsson (2022),

[1] `https://github.com/huggingface/tokenizers`

[2] `https://github.com/tensorflow/models`

we performed a limited hyperparameter search to identify a single robust configuration applicable to all models and languages. Table 2.1 summarizes these hyperparameter settings.

For POS, NER, QA, and TC, our models were fine-tuned using the Transformers library for Python (Wolf et al., 2020). We used task-specific fine-tuning scripts for PyTorch (Ansel et al., 2024) included with the library: `run_ner.py` for POS and NER, `run_qa.py` for QA, and `run_classification.py` for TC.

For POS tagging and NER, we followed Dađason and Loftsson (2022) and fine-tuned the models for 20 and 10 epochs, respectively, using a batch size of 16 and a learning rate of 5e-5. We reported tagging accuracy for POS tagging and entity-level F_1 scores for NER.

For QA and TC, which were not evaluated by Dađason and Loftsson (2022), we performed a limited hyperparameter search for learning rates in $[3e{-}5, 5e{-}5, 8e{-}5, 1e{-}4]$, batch sizes in $[8, 16, 32]$, and number of epochs in $[5, 10, 20]$. Based on this search, we fine-tuned QA models for 5 epochs with a learning rate of 8e-5 and a batch size of 16, reporting F_1 scores. For TC, we selected 10 epochs, a learning rate of 1e-4, and a batch size of 8, reporting classification accuracy.

For dependency parsing (DP), we used DiaParser (Attardi et al., 2021), a biaffine dependency parser that extracts contextual word embeddings from Transformer-based LMs. Models were fine-tuned for 200 epochs with a batch size of 5,000 and a learning rate of 2e-3. The model checkpoint that achieved the highest labeled attachment score (LAS) on the validation set was selected and evaluated on the test set.

For automatic text summarization (ATS), we used TransformerSum[3], a Python library based on the BertSum extractive text summarization model (Liu and Lapata, 2019). Models were fine-tuned for 3 epochs with a batch size of 8 and a learning rate of 2e-5. We evaluated performance using the ROUGE metric, which measures the ratio of overlapping n-grams between the target summary and the generated summary. Specifically, we reported ROUGE-2 recall scores. We used mean pooling to generate sentence-level embeddings, and applied a linear classifier to select sentences for the predicted summary. For each document, we selected sentences for the predicted summary until it contained at least 100, then computed the ROUGE score by comparing it to the target summary. Like Dađason et al. (2021), when creating the training data, we used an oracle to label each sentence in the original document, greedily maximizing the ROUGE-2 recall score until the summary contained at least 100 words.

Task	Metric	Batch Size	Learning Rate	Epochs
POS	Accuracy	16	5e-5	20
NER	F_1 Score	16	5e-5	10
DP	LAS	5000	2e-3	200
ATS	ROUGE-2 Recall	8	2e-5	3
QA	F_1 Score	16	8e-5	5
TC	Accuracy	8	1e-4	10

Table 2.1: Hyperparameters for fine-tuning TEAMS-Small models.

The evaluation datasets are described in Section 3.3. For each dataset, we generally evaluated the models using the provided splits. For datasets with official training,

[3] `https://github.com/HHousen/TransformerSum`

validation, and test sets, we reported average scores over 10 runs with different random seeds. For datasets distributed with a k-fold split, we evaluated the model using k-fold cross-validation. If no splits were provided with a dataset, we performed stratified 10-fold cross-validation. We made an exception to these rules for datasets that contained fewer than 10,000 labeled examples, where we performed stratified 5-fold cross-validation with 5 repetitions to reduce variance in scores between different folds or runs and to increase statistical power for significance testing.

When comparing the downstream performance of two models, we performed a paired t-test to determine if there was a statistically significant difference between their results. For significance testing on more than two models, we additionally performed Holm-Bonferroni correction, a method used to control the family-wise error rate in multiple comparisons.

Chapter 3

Languages and Resources

In this chapter, we describe the languages and language resources used in our experiments, which form the foundation for addressing our research questions. We begin with an overview of the six languages we selected, discussing their linguistic characteristics and the state of language technology (LT) for each. Next, we describe the unannotated corpora that we used, detailing their size, sources, and other properties. Finally, we outline the annotated datasets used for evaluating the downstream performance of pre-trained language models (LMs), which provide a benchmark for assessing the effectiveness of different pre-training techniques.

3.1 Language Selection

To answer our research questions, we selected six low to medium-resource languages: Icelandic, Estonian, Basque, Galician, Nepali, and Tajik. These languages were chosen for their diverse linguistic characteristics, including different language families, morphological typologies, and levels of language resource availability.

Icelandic is a North Germanic language within the Indo-European family, spoken by approximately 330,000 native speakers[1]. It is a highly inflectional language with rich morphology. Icelandic has an established National LT Programme (Nikulásdóttir et al., 2020; Nikulásdóttir et al., 2022), which has, during the last few years, significantly advanced its LT infrastructure.

Estonian is a Finnic language in the Uralic family, with 1.2 million native speakers. Its predominantly agglutinative morphology includes some fusional features. Like Icelandic, Estonian has a well-developed National LT Programme (Vider et al., 2012).

Basque is a language isolate, unrelated to any known language, with about 800,000 native speakers. Its agglutinative morphology and ergative-absolutive alignment make it typologically distinct. Basque is an official language in the Basque Country and Basque-speaking areas of Navarre, autonomous regions of Spain. The development of LT in Basque Country has quite a long history (Alegria and Sarasola, 2017).

Galician is an Ibero-Romance language in the Indo-European family, closely related to Portuguese. It has 2.4 million native speakers and exhibits fusional morphology. Galician is an official language in the autonomous region of Galicia, Spain, and has benefited from national LT initiatives, including an established National LT Programme (de Dios-Flores et al., 2022). Moreover, national research initiatives have supported LT infrastructure development for minority languages across Spain (Agerri et al., 2018a).

[1]Native speaker statistics are based on Wikipedia, which references Ethnologue.

Nepali is an Indo-Aryan language in the Indo-European family, primarily spoken in the Himalayan region of South Asia. It uses the Devanagari script and primarily exhibits fusional morphology, with some agglutinative features. Nepali has 19 million native speakers and is an official language of Nepal and certain Indian regions. Despite its large speaker base, LT resources for Nepali remain scarce. Nepal is classified as a least developed country by the United Nations, and while it lacks a National LT Programme, international research collaborations such as Nelralec (the Bhasa Sanchar Project) have produced resources like the Nepali National Corpus (NNC) (Yadava et al., 2008).

Tajik is a Western Iranian language in the Indo-European family, closely related to Persian, but written in Cyrillic script. Its morphology is moderately fusional with analytic tendencies. Tajik has 10 million native speakers and is an official language in the developing country of Tajikistan. No National LT Programme exists for Tajik, and language resources are limited.

While historically considered under-resourced, Icelandic, Estonian, Basque, and Galician have seen substantial development in their LT infrastructure over the past decade. National LT Programmes have been established for Icelandic, Estonian, and Galician, and significant progress has been made in LT for Basque through regional research initiatives. In contrast, despite having relatively large speaker populations, Nepali and Tajik remain notably under-resourced, with limited digital infrastructure and fewer available language resources.

There is no universally accepted definition of what separates low-, medium-, and high-resource languages. In this thesis, we define resource levels based on two key criteria: (1) the availability of a high-quality monolingual corpus of sufficient size for effective LM pre-training, which empirical evidence suggests is at least 100 MB of text (Micheli et al., 2020), and (2) the availability of manually annotated, monolingual datasets for core natural language processing (NLP) tasks, such as part-of-speech (POS) tagging, named entity recognition (NER), and dependency parsing (DP). A language is classified as low-resource if it fails to meet either of these criteria, and medium-resource if it meets both. In contrast, high-resource languages typically have corpora amounting to hundreds of billions of tokens, along with a significantly broader range of datasets for more advanced NLP tasks suitable for evaluating large-scale LMs.

Our selection includes both low-resource languages (Nepali, Tajik) and medium-resource languages (Icelandic, Estonian, Basque, Galician), representing a variety of morphological typologies–from fusional (Icelandic, Galician, Nepali, Tajik) to agglutinative (Estonian, Basque). These languages also represent three different writing systems: Latin, Cyrillic, and Devanagari. We conclude that this diverse selection of morphologically rich languages offers strong evidence to address our research questions.

3.2 Corpora

This section describes the unannotated corpora that we used in our experiments, which include both high-quality, curated corpora, as well as corpora derived from web content. The web-based corpora either originate from large-scale web archives, such as Common Crawl (CC)[2], or from smaller-scale, targeted web scraping efforts with ad-hoc crawlers.

For each corpus, we report its size in terms of uncompressed disk space (in gigabytes or megabytes), the number of documents, and the number of space-delimited tokens

[2] `https://commoncrawl.org/about/`

(i.e., strings separated by whitespace characters). Table 3.1 provides an overview of these statistics for each language and corpus.

Language	Corpus	Size (GB)	Documents	Tokens
Icelandic	IC3	4.89	2,169,989	824,301,374
Icelandic	ICC	4.79	1,934,732	817,261,940
Icelandic	IGC	10.73	5,125,016	1,674,902,856
Icelandic	mC4	7.70	2,069,293	1,121,745,925
Estonian	ENC	3.14	2,143,795	428,967,956
Estonian	mC4	22.50	6,941,360	3,044,585,733
Basque	EusCrawl	2.11	1,585,430	288,324,729
Basque	mC4	4.31	1,555,887	575,606,522
Galician	mC4	7.54	4,549,465	1,209,371,725
Galician	Wikipedia	0.44	199,789	73,450,656
Nepali	mC4	17.10	2,942,785	967,304,877
Nepali	Wikipedia	0.09	32,572	5,809,035
Tajik	mC4	6.99	1,280,757	561,480,539
Tajik	Wikipedia	0.12	106,185	10,220,523

Table 3.1: Overview of corpora used in our experiments. The table includes corpus size in GB (uncompressed text, without metadata), the number of documents, and the number of space-delimited tokens. Note that the statistics for the IGC and ENC corpora exclude subcorpora with web-crawled content or text shuffled on the sentence or paragraph level.

The curated corpora, such as IGC, ENC, EusCrawl, and Wikipedia, represent high-quality content, containing text from carefully chosen sources. In contrast, web-crawled corpora like mC4, IC3, and ICC, encompass large-scale, diverse content, but may vary in quality and require additional pre-processing. This balance of curated and web-crawled corpora allows us to gain insight into the impact of data quality and diversity on the downstream performance of pre-trained LMs.

3.2.1 Icelandic Corpora

For Icelandic, we used three corpora: the Icelandic Gigaword Corpus (IGC), the Icelandic Crawled Corpus (ICC), and the Icelandic Common Crawl Corpus (IC3), each of which is described below.

The Icelandic Gigaword Corpus (IGC)

The IGC[3] (Steingrímsson et al., 2018) contains approximately 2.3 billion tokens from diverse Icelandic text sources, including news articles, parliamentary speeches, and adjudications. We used the 2022 version of the corpus but omitted subcorpora that we deemed unsuitable for pre-training LMs. Specifically, we excluded the following subcorpora:

- **Online forum discussions, journal articles, and published books**: To comply with licensing agreements and copyright restrictions, these subcorpora

[3]IGC: `https://igc.arnastofnun.is/`

are presented as shuffled sentences or paragraphs, disrupting the natural context and long-range dependencies essential for learning meaningful relationships and representations.

- **Twitter posts**: These social media posts are included in the corpus as unique identifiers for downloading the original content. These posts are typically very short and lack the coherence and quality of the more curated subcorpora, making them less suitable for pre-training LMs.

After excluding these subcorpora, the remaining subcorpus contains approximately 1.7 billion space-delimited tokens across 5.1 million documents. Table 3.2 provides an overview of the IGC subcorpora (after excluding the two subcorpora mentioned above), including their size, document count, and number of tokens.

Subcorpus	Size (GB)	Documents	Tokens
Adjudications	0.45	29,781	68,929,493
Books	0.09	398	14,103,849
Journals	0.14	20,932	20,754,577
Law	0.36	16,367	53,122,049
News 1	2.57	1,781,331	394,061,394
News 2	5.70	3,196,233	888,910,499
Parliamentary Speeches	1.56	19,252	254,879,866
Social Media - Blogs	0.06	38,342	8,987,964
Social Media - Forums	3.20	718	561,021,669
Wikipedia	0.06	54,649	8,374,587
Total	14.19	5,158,003	2,273,145,947

Table 3.2: Statistics for the IGC and its subcorpora, including size in GB (uncompressed text, without metadata), number of documents, and space-delimited tokens. The social media subcorpus is broken down into blogs and forum posts, but Twitter posts are omitted.

The Icelandic Crawled Corpus (ICC)

The ICC[4] was compiled (by the author of this thesis) by scraping text from 24 Icelandic websites. It contains approximately 817 million space-delimited tokens, drawn from online forum posts (772M tokens), news articles (27M tokens), adjudications (10M tokens), and blog posts (8M tokens). The websites were scraped using ad-hoc crawlers, with text extraction performed using the Beautiful Soup library for Python (Richardson, 2020). Custom extraction rules were implemented for each domain to target relevant textual content while discarding boilerplate text (e.g., headers, footers, and metadata), advertisements, and other unrelated elements. Exact duplicate documents were removed, but no additional filtering was applied to improve the quality of the corpus.

[4]ICC: `https://huggingface.co/datasets/jonfd/ICC`

The Icelandic Common Crawl Corpus (IC3)

The IC3[5] (Snæbjarnarson, 2021) was derived from the CC dataset, consisting of documents archived from 2008 to 2020 from websites with the Icelandic top-level domain (.is). Plain text was extracted using jusText (Pomikálek, 2011), which removed boilerplate content such as advertisements, headers, and footers. To improve data quality, a fastText (Bojanowski et al., 2017) language classifier was applied to retain only documents with Icelandic as the primary language. Additionally, exact duplicate documents and repeated three-line spans were removed. The dataset is distributed in a pre-tokenized format, with text already segmented into words and punctuation marks separated from words. After these filtering steps, the IC3 corpus contains approximately 824 million tokens across 2.2 million documents.

3.2.2 Estonian Corpora

For Estonian, we used the Estonian National Corpus (ENC), a comprehensive dataset of written Estonian texts. Below, we describe its structure and contents.

Estonian National Corpus (ENC)

The ENC[6] (Koppel and Kallas, 2022) consists of approximately 2.5 billion space-delimited tokens, making it one of the most comprehensive collections of Estonian written language. We used the 2021 version of the corpus, which combines both web-crawled and curated texts from a wide range of genres, representing Estonian written language from the 1990s to 2021.

Approximately 83% of the ENC consists of web-crawled content across four collection periods: Web 2013, 2017, 2019, and 2021. These texts primarily include periodicals, forum posts, and blogs. The remaining 17% comprises curated texts, including news feeds, literary works, academic writing, and Wikipedia articles, which offer higher-quality and domain-specific content.

We chose to omit the web-crawled portion of the ENC for our experiments and instead used only its curated subcorpora. This decision was motivated by the need to better evaluate the impact of data quality on the downstream performance of pretrained LMs. For Estonian, we used the curated portion of the ENC as a source of high-quality content, while using the mC4 corpus (described in Section 3.2.4) for noisy, web-crawled data. Unlike the web-crawled portion of the ENC, which has undergone extensive filtering to improve text quality, the mC4 corpus retains much of the noise typical of raw web content. This should provide us with more conclusive evidence for answering our research questions.

Table 3.3 provides an overview of the ENC subcorpora, including their size, document count, and number of tokens.

3.2.3 Basque Corpora

For Basque, we used the EusCrawl corpus, a high-quality collection of texts derived from carefully selected Basque-language websites. Below, we provide a description of its structure.

[5]IC3: `https://huggingface.co/datasets/mideind/icelandic-common-crawl-corpus-IC3`
[6]ENC: `https://doi.org/10.15155/3-00-0000-0000-0000-08D17L`

Subcorpus	Size (GB)	Documents	Tokens
Balanced Corpus	0.07	21,688	9,780,089
News Feeds	1.45	1,262,343	197,218,554
Fiction	0.11	203	16,703,052
Open Access Journals	0.07	1,831	9,176,586
Reference Corpus	1.33	716,564	180,945,388
Web 2013	1.84	681,933	257,627,961
Web 2017	3.86	1,913,160	540,380,782
Web 2019	3.66	2,464,244	517,530,949
Web 2021	5.29	4,545,454	743,368,560
Wikipedia	0.07	102,088	9,197,224
Wikipedia Talk	0.04	39,078	5,947,063
Total	17.79	11,748,586	2,487,876,208

Table 3.3: Statistics for the ENC and its subcorpora, including size in GB (uncompressed text, without metadata), number of documents, and space-delimited tokens.

EusCrawl

The EusCrawl corpus[7] (Artetxe et al., 2022) consists of 288 million space-delimited tokens across 1.58 million documents. It was created by scraping content from 33 high-quality Basque-language websites, primarily in the news domain.

The quality of the EusCrawl corpus was benchmarked by Artetxe et al. (2022) against other web-crawled corpora, including mC4 (Xue et al., 2021) and CC-100 (Conneau et al., 2020). Annotators manually reviewed 100 randomly selected documents from each corpus, evaluating them with regard to language identification, correctness, coherence, noise, content quality, and overall suitability for inclusion in the corpus. Approximately $\frac{2}{3}$ of the documents in EusCrawl were rated as high quality, significantly outperforming mC4 and CC-100, where less than a third of the documents met the same standard. This high-quality content makes EusCrawl particularly well-suited for pre-training LMs for the Basque language.

3.2.4 Multilingual Corpora

We used two multilingual corpora: the Multilingual Colossal Clean Crawled Corpus (mC4) and Wikipedia. The mC4 corpus consists of large-scale web-crawled text with minimal filtering, while Wikipedia contains high-quality encyclopedic articles. Below, we describe each corpus.

The Multilingual Colossal Clean Crawled Corpus (mC4)

The mC4[8] (Xue et al., 2021) is a large-scale dataset created by extracting documents from the entire Common Crawl dataset and classifying them by their primary language. It includes subsets for 108 languages, with languages containing fewer that 10,000 documents excluded.

[7]For EusCrawl, we used the plain text version of the corpus distributed at `https://www.ixa.eus/euscrawl/`.

[8]mC4: `https://huggingface.co/datasets/allenai/c4`

To improve text quality, the authors employed basic filtering techniques, such as removing duplicate three-line spans and discarding lines that do not end with punctuation marks. While this corpus retains significant diversity and scale, the minimal amount of filtering performed can lead to noisy or lower-quality text. Originally created to pre-train the multilingual mT5 language model, mC4 serves as a valuable resource for pre-training LMs on large-scale data, especially for less-resourced languages where curated corpora are not readily available.

Table 3.4 provides an overview of the subsets of the mC4 corpus for each language, including their size, document count, and number of tokens.

Language	Size (GB)	Documents	Tokens
Icelandic	7.70	2,069,293	1,121,745,925
Estonian	22.50	6,941,360	3,044,585,733
Basque	4.31	1,555,887	575,606,522
Galician	7.54	4,549,465	1,209,371,725
Nepali	17.10	2,942,785	967,304,877
Tajik	6.99	1,280,757	561,480,539

Table 3.4: Statistics for each subset of the mC4 corpus, including size in GB (uncompressed text, without metadata), number of documents, and space-delimited tokens.

Wikipedia

Wikipedia is a widely used, crowdsourced online encyclopedia that provides high-quality text for many low- and medium-resource languages. We used cleaned versions of Wikipedia articles obtained from Wikimedia's Hugging Face repository[9], where markdown and unwanted sections such as references had been removed.

Language	Size (MB)	Documents	Tokens
Galician	453.5	199,789	73,450,656
Nepali	97.2	32,572	5,809,035
Tajik	120.0	106,185	10,220,523

Table 3.5: Statistics for each Wikipedia corpus, including size in MB (uncompressed text, without metadata), number of documents, and space-delimited tokens.

3.3 Evaluation Datasets

To address our research questions, we fine-tune and evaluate each pre-trained LM on a diverse benchmark of downstream tasks. These tasks include POS tagging, NER, DP, topic classification (TC), automatic text summarization (ATS), and question answering (QA). Table 3.6 provides an overview of the datasets, their associated tasks, and the number of labeled examples.

We use the Universal Dependencies (UD) dataset (Nivre et al., 2016) for evaluating the LMs on DP in Icelandic, Estonian, Basque, and Galician, as well as for POS tagging

[9] `https://huggingface.co/datasets/wikimedia/wikipedia`

in Estonian and Basque. UD is a comprehensive multilingual dataset, which, as of version 2.14, contains 283 treebanks in 161 languages, making it particularly valuable for low- and medium-resource languages. The treebanks are annotated with universal POS (UPOS) tags, with some being natively annotated in the UD format and others converted from existing treebanks with different annotation schemes to conform to UD guidelines. We used version 2.14 of the UD dataset[10].

In addition to UD, we employ a variety of task-specific datasets to evaluate downstream performance in each language. These include datasets for POS (MIM-GOLD for Icelandic and CTG for Galician), NER (e.g., MIM-GOLD-NER for Icelandic and EstNER-New for Estonian), TC (SIB-200 for Nepali and Tajik), and ATS (IceSum for Icelandic). Dataset sizes are reported as the number of labeled examples.

Language	Task	Dataset	Size
Icelandic	POS	MIM-GOLD	1,000,218
Icelandic	NER	MIM-GOLD-NER	1,000,231
Icelandic	DP	IcePaHC-UD	983,668
Icelandic	ATS	IceSum	1,000
Icelandic	QA	RUQuAD	10,530
Estonian	POS	EDT-UD	437,769
Estonian	NER	EstNER-New	184,638
Estonian	DP	EDT-UD	437,769
Basque	POS	BDT-UD	121,443
Basque	NER	EIEC	59,759
Basque	DP	BDT-UD	121,443
Galician	POS	CTG	1,196,734
Galician	NER	NERC	202,334
Galician	DP	Galician-TreeGal-UD	23,479
Nepali	POS	NNC-CS	1,123,528
Nepali	NER	EverestNER	308,353
Nepali	TC	SIB-200	1,004
Tajik	NER	WikiANN	300
Tajik	TC	SIB-200	1,004

Table 3.6: Languages, tasks, and datasets in the evaluation benchmark, including dataset size measured in number of labeled examples.

3.3.1 Icelandic Datasets

To evaluate Icelandic LMs, we used five task-specific datasets: MIM-GOLD for POS tagging, MIM-GOLD-NER for NER, IcePaHC-UD for DP, IceSum for extractive ATS, and RUQuAD for QA. Below, we provide descriptions of each dataset.

MIM-GOLD

MIM-GOLD (Loftsson et al., 2010) is a subset of the IGC which has been semi-automatically labeled with POS tags. We used version 21.05 of the corpus[11], which

[10]UD v2.14: `http://hdl.handle.net/11234/1-5502`

[11]MIM-GOLD: `http://hdl.handle.net/20.500.12537/114`

includes 1,000,218 tokens across 58,412 sentences, annotated with a total of 557 distinct tags.

MIM-GOLD-NER

MIM-GOLD-NER (Ingólfsdóttir et al., 2020) is a version of MIM-GOLD that has been semi-automatically labeled with named entities in the IOB2 format. In this tagging scheme, each token is assigned a label indicating whether it is part of a named entity and, if so, its position within the entity. The first token of an entity is tagged with a B- prefix (beginning), while subsequent tokens of the same entity are tagged with I- (inside). Tokens that do not belong to any named entity are labeled O (outside). The dataset includes eight entity types: person, location, organization, miscellaneous, date, time, money, and percent. We used version 2.0 of MIM-GOLD-NER[12], which contains a total of 47,764 named entities.

Icelandic Parsed Historical Corpus (IcePaHC)

IcePaHC (Rögnvaldsson et al., 2012) consists of one million tokens, from texts spanning the 12^{th} century to modern times, that have been manually annotated with constituents, which are words or groups of words that function as a single unit within a syntactic tree, such as noun phrases (NP), verb phrases (VP), or clauses. We used a version of IcePaHC which has been converted to the UD format[13] (Arnardóttir et al., 2020).

IceSum

IceSum[14] (Daðason et al., 2021) is a dataset of 1,000 news articles in Icelandic which have been manually annotated with extractive summaries. The articles, published between 1998 and 2019, consist of four categories: local news (50%), world news (26%), business (14%) and sports news (10%). The average article length is 302 words, while the summaries average 102 words.

Reykjavik University Question-Answering Dataset (RUQuAD)

RUQuAD[15] (Skarphéðinsson et al., 2023) is dataset that contains approximately 20,800 questions and 12,700 answers, with some questions having one or more answers, and others having none. The dataset was constructed using spurningar.is, a crowdsourcing app where users generated and reviewed questions, selected paragraphs with the correct answer through an online search, and identified the exact answer spans. RUQuAD contains 10,530 distinct questions that have one or more answer.

3.3.2 Estonian Datasets

To evaluate Estonian LMs, we used two datasets: EDT-UD for POS tagging and DP, and EstNER-New for NER. Below, we describe each dataset.

[12]MIM-GOLD-NER: `http://hdl.handle.net/20.500.12537/230`

[13]`https://universaldependencies.org/treebanks/is_icepahc/index.html`

[14]IceSum: `http://hdl.handle.net/20.500.12537/285`

[15]RUQuAD: `http://hdl.handle.net/20.500.12537/310` and `http://hdl.handle.net/20.500.12537/311`

Estonian Dependency Treebank (EDT)

The EDT (Muischnek et al., 2014) consists of a subset of the Estonian Reference Corpus which has been annotated with syntactic dependencies. The texts in the EDT span various genres, including newspaper texts, fiction, and scientific texts. For our experiments, we used the UD version of EDT (EDT-UD)[16] (Muischnek et al., 2016), which contains 438,245 tokens across 30,968 sentences, annotated with syntactic dependencies and 16 distinct UPOS tags.

EstNER-New

The New EstNER corpus[17] (Sirts, 2023) contains 139,674 tokens across 8,773 sentences that have been manually labeled with named entities. The dataset uses the IOB2 annotation format and includes 11 named entity types: person, organization, location, geopolitical entity, title, product, event, date, time, percentage, and monetary value.

EstNER was annotated with nested entities, where "New York City Government" might be annotated as an organization, while "New York" would be labeled as a geopolitical entity. The nesting was limited to three levels of depth. The annotation was performed by 12 native Estonian speakers, with each token receiving three independent labels. For the outermost entities, the annotators obtained a substantial inter-agreement rate of 0.65 using Fleiss' Kappa (Fleiss, 1971).

For our experiments, we flattened the dataset and used only the outermost entity annotations, resulting in a total of 9,632 named entities. This allowed us to evaluate the EstNER-New dataset in the same manner as the other NER datasets, which do not feature nested annotations.

3.3.3 Basque Datasets

To evaluate Basque LMs, we used two datasets: BDT-UD for POS tagging and DP, and EIEC for NER. Below, we provide a description of each dataset.

Basque Dependency Treebank (BDT)

The BDT (Aduriz et al., 2003) was initially composed of approximately 50,000 words from newspaper texts and other genres, manually annotated with syntactic dependencies. For our experiments, we used the UD version of the BDT (BDT-UD)[18] (Aranzabe et al., 2015), which has been expanded to include 121,443 tokens across 8,993 sentences. Additionally, BDT-UD has been annotated with morphosyntactic features, with each token assigned one of 17 distinct UPOS tags.

Basque Named Entities Corpus (EIEC)

The EIEC[19] (Alegria et al., 2004) consists of newswire articles that have been semi-automatically annotated with named entities. This corpus contains 59,759 tokens across 3,394 sentences, annotated in the IOB2 format with four named entity types: person, location, organization, and other.

[16]EDT: `https://universaldependencies.org/treebanks/et_edt/index.html`
[17]EstNER-New: `https://github.com/TartuNLP/EstNER_new`
[18]BDT-UD: `https://universaldependencies.org/treebanks/eu_bdt/index.html`
[19]EIEC: `http://ixa2.si.ehu.eus/eiec/eiec_v1.0.tgz`

3.3.4 Galician Datasets

To evaluate Galician LMs, we used three datasets: CTG-POS for POS tagging, NERC for NER, and Galician-TreeGal for DP. Descriptions of each dataset are provided below.

Galician Technical Corpus (CTG)

The CTG (Agerri et al., 2018b) contains 18 million words drawn from technical domains such as computing, economics, and medical texts. For POS tagging, we used a manually annotated subset of this corpus, CTG-POS. Specifically, we used version 1.1 of the CTG-POS[20], which consists of 1,196,734 tokens annotated with 183 distinct POS tags.

SLI NERC Galician Gold Corpus (NERC)

The SLI NERC Galician Gold Corpus[21] (Agerri et al., 2018b) was created by manually annotating a subset of the CTG corpus with named entities. This dataset consists of texts from the news and ecology/environmental sciences domains. It includes 8,467 named entities annotated across 8,137 sentences and 202,334 tokens. The annotation follows the IOB2 format, with four entity types: person, organization, location, and miscellaneous.

Galician-TreeGal

The Galician-TreeGal corpus[22] (Garcia et al., 2018) consists of 23,479 tokens across 1,000 sentences of newspaper text. The dataset has been semi-automatically annotated with syntactic dependencies in the UD format.

3.3.5 Nepali Datasets

To evaluate Nepali LMs, we used two monolingual datasets: NNC-CS for POS tagging and EverestNER for NER. These datasets are described below. Additionally, Nepali LMs were evaluated on the multilingual SIB-200 dataset for TC, as described in Section 3.3.6.

Nepali National Corpus (NNC)

The NNC[23] (Yadava et al., 2008) contains 14 million words of written and spoken Nepali, drawn from genres such as news articles, essays, and fiction. All texts in the NNC were originally published in 1991. A curated subset of this corpus, referred to as the NNC Core Sample (NNC-CS), consists of approximately 800,000 words and is designed to be balanced and representative.

The NNC-CS underwent POS tagging using a bootstrap approach. Initially, around 300,000 tokens were manually annotated, which were then used to train a hybrid rule-based and probabilistic tagger. This tagger was subsequently used to annotate the

[20]CTG-POS: `https://github.com/xavier-gz/SLI_Galician_Corpora`

[21]NERC: `https://github.com/xavier-gz/SLI_Galician_Corpora`

[22]Galician-TreeGal: `https://universaldependencies.org/treebanks/gl_treegal/index.html`

[23]NNC: `https://catalogue.elra.info/en-us/repository/browse/ELRA-W0076/`

remaining corpus. The authors report an accuracy of 93% for the tagger. However, the published version of the corpus does not differentiate between manually and automatically tagged texts. The final NNC-CS dataset includes 1,123,528 tokens across 59,668 sentences, annotated with 116 distinct POS tags.

EverestNER

EverestNER[24] (Niraula and Chapagain, 2022) consists of 996 Nepali news articles that have been manually annotated with named entities. Two annotators labeled the dataset in the IOB2 format using five entity categories: person, location, organization, event, and date. The inter-annotator agreement rate was substantial at 0.74, measured using Cohen's Kappa (McHugh, 2012). A total of 24,587 entities were annotated across 15,798 sentences and 308,353 tokens.

3.3.6 Multilingual Datasets

We included two multilingual datasets in our evaluation: SIB-200, used to evaluate Nepali and Tajik on TC, and WikiANN, used to evaluate Tajik on NER. Descriptions of these datasets are provided below.

SIB-200

SIB-200[25] (Adelani et al., 2024) is a topic classification dataset derived from the FLORES-200 corpus (NLLB Team et al., 2022). The FLORES-200 corpus consists of 3,001 sentences, originally sampled from English-language Wikimedia projects and professionally translated into 203 additional languages. To create SIB-200, 1,004 sentences were sampled from FLORES-200, and each English sentence was manually annotated with one of seven topic labels: science/technology, travel, politics, sports, health, entertainment, or geography. These labels were then propagated to all translated languages. The distribution of topic classes in SIB-200 is imbalanced, with category sizes ranging from 83 to 252 sentences.

WikiANN

WikiANN (Pan et al., 2017) is a multilingual corpus for NER, consisting of sentences from 282 languages extracted from Wikipedia. The sentences were automatically annotated with named entities using the IOB2 format, with three entity types: person, location, and organization. For our experiments, we used a version of the dataset published by Rahimi et al. (2019), which provides a balanced subset of the original WikiANN corpus[26].

3.4 Other Datasets

Below, we describe additional datasets and resources that were used for our experiments.

[24]EverestNER: `https://github.com/nowalab/everest-ner`
[25]SIB-200: `https://huggingface.co/datasets/Davlan/sib200`
[26]WikiANN: `https://huggingface.co/datasets/unimelb-nlp/wikiann`

Stop Word Lists

Stop words are commonly occurring words, such as function words (e.g., articles, pronouns, and prepositions), that primarily serve grammatical functions rather than carrying significant meaning on their own. Their role in NLP depends on the task at hand; for example, they are often filtered out in search engines to reduce noise, while they may be retained as useful features in tasks like language classification. In our experiments, stop word lists were used for text quality filtering (see Section 4.3), where we investigated their effectiveness as indicators of high-quality text in web-crawled corpora. We obtained stop word lists for each language from the following publicly available sources:

- **Icelandic**: We used a list of stop words prepared by Atli Jasonarson[27], which contains 590 stop words extracted from the Database of Icelandic Morphology (DIM) (Bjarnadóttir et al., 2019).
- **Estonian**: We used a list of stop words created by Kristel Uiboaed and distributed on the University of Tartu data repository[28]. This list contains 5,025 unique inflectional forms representing 1,605 lemmas.
- **Basque and Galician**: Stop word lists were obtained from the Stopwords ISO collection[29], containing 98 and 160 unique stop words, respectively.
- **Nepali and Tajik**: For these languages, we used stop word lists provided through the NLTK library for Python[30] (Bird et al., 2009). These lists contain 254 and 218 stop words, respectively.

[27]Icelandic stop words: `https://github.com/atlijas/icelandic-stop-words`
[28]Estonian stop words: `https://datadoi.ee/handle/33/78`
[29]Basque and Galician stop words: `https://github.com/stopwords-iso`
[30]Nepali and Tajik stop words: `https://www.nltk.org/nltk_data/`

Chapter 4

Text Filtering

Early Transformer-based language models (LMs), such as GPT (Radford et al., 2018) and BERT (Devlin et al., 2019), were typically pre-trained on curated corpora consisting of up to several billion words. It is now well established that increasing the size of pre-training corpora can significantly improve the downstream performance of such models (Liu et al., 2019). Consequently, it has become common practice to supplement high-quality pre-training corpora with, or even rely exclusively on, documents scraped from online sources (Brown et al., 2020; Raffel et al., 2020; Xue et al., 2021; Wu et al., 2021). Online texts are often obtained from large datasets, such as those published by Common Crawl (CC).[1] For instance, the GPT-3 model was pre-trained on 499 billion tokens, 410 billion of which were obtained from CC (Brown et al., 2020), while the T5 model was pre-trained on one trillion tokens from CC (Raffel et al., 2020).

Although web-crawled corpora offer researchers the opportunity to dramatically increase the size of their datasets, they are typically quite noisy. These corpora often contain significant amounts of low-quality text, such as HTML tags, JavaScript code, navigation menus, headers, footers, boilerplate text, text in unwanted languages, or text that is otherwise incoherent or incomplete. An audit of 205 web-crawled corpora revealed that the ratio of usable text was below 50% in 87 of them, with 15 corpora containing no usable text at all (Kreutzer et al., 2022). The exact definition of "noisy" or "low-quality" text varies and is subject to interpretation. In the context of LM pre-training, text quality can be understood in terms of its impact on model learning. High-quality text helps models learn useful syntactic and semantic information that transfers well to downstream tasks, while low-quality text may contribute noise to the learning process or even degrade model performance. It has been repeatedly demonstrated that filtering web-crawled corpora can significantly improve the downstream performance of pre-trained LMs (Brown et al., 2020; Raffel et al., 2020; Muennighoff et al., 2023).

Filtering is typically performed using heuristic rules or classifiers. In the rule-based approach, documents are filtered out if certain metrics, such as mean word length or stop word ratio, fall outside a predefined acceptable range (Rae et al., 2022). There is no standardized approach to rule-based text quality filtering: some corpora are filtered using only a single rule (Wenzek et al., 2020), while others combine up to 15 distinct rules (Öhman et al., 2023). As the size of the ruleset increases, it can become more difficult to determine the impact that individual rules might have on the overall effectiveness of the filtering process. Rules that may be effective individually may become redundant as more rules are added. Conversely, a rule that appears ineffective

[1] `https://commoncrawl.org/about/`

on its own may become more useful when applied in conjunction with others. The approaches to selecting the thresholds for the rules also vary significantly. Thresholds are often chosen based on linguistic intuition (Rae et al., 2022; Laurençon et al., 2022; Öhman et al., 2023) or statistical analysis, such as having each rule discard a fixed percentage of documents in the corpus (Nguyen et al., 2023; Young et al., 2024).

Alternatively, classifiers can be used to label or score documents based on their quality. This includes supervised classifiers, trained on a manually labeled text quality dataset (Wu et al., 2021), and weakly supervised classifiers, trained to distinguish between documents from a high-quality, curated corpus and a noisy, web-crawled corpus as a proxy for text quality (Brown et al., 2020). The effectiveness of text quality classifiers depends on the choice of features, parameters, training data, and model type. The parameters may be chosen through statistical analysis, such as aligning the distribution of the filtered corpus with that of a known high-quality corpus (Brown et al., 2020).

In either case, the quality of the chosen thresholds (in the case of rules) or parameters (in the case of classifiers) can only be assessed through empirical validation. This may involve manually labeling a representative sample for evaluation (Wu et al., 2021), or comparing the downstream performance of LMs pre-trained on filtered and unfiltered versions of the corpus (Raffel et al., 2020). Although text filtering is standard practice when relying on web-crawled text, detailed, fine-grained evaluations of individual filters remain relatively uncommon. When evaluations are performed, they often measure the overall impact of text filters, which often comprise multiple heuristic rules or text quality classifiers (e.g., the work by Raffel et al. (2020); Rae et al. (2022)), rather than evaluating the impact of each individual filter. Furthermore, detailed information on training data and experimental settings for text quality classifiers is sometimes omitted (e.g., the work by Wu et al. (2021)). However, some recent work has begun to address this gap, such as Penedo et al. (2024), which systematically examines individual filtering and deduplication choices for English.

In this chapter, we evaluate the effectiveness of both rule-based and classifier-based approaches to text quality filtering. We begin by describing TQ-IS, a new, manually annotated text quality dataset for Icelandic. To the best of our knowledge, TQ-IS is the first publicly available dataset designed specifically for document-level text quality classification, enabling both direct training and evaluation of document-level classifiers. Using TQ-IS, we evaluate the effectiveness of numerous heuristic rules commonly used for text filtering and three text quality classifiers based on previously described approaches. Additionally, we propose a novel text quality classifier by reframing the task as an outlier detection problem, with low-quality documents as outliers. We evaluate three types of clustering and outlier detection algorithms on TQ-IS. The main benefit of these algorithms is their unsupervised nature, explainability, and the fact that only few parameters, none of which require language expertise, need to be tuned. Finally, we use the best performing method to filter web-crawled corpora in six low to medium-resource languages, and evaluate how the filtering step impacts the downstream performance of pre-trained LMs for those languages. This experiment aims to answer our first research question: **RQ1: How do different text filtering techniques impact the downstream performance of LMs pre-trained on web-crawled corpora for low- and medium-resource languages?**

Filtering can be applied at various levels of granularity, from tokens and sentences to paragraphs and entire documents. While more granular filtering can potentially preserve more usable text, it introduces several challenges. Sentence- and paragraph-

level filtering depends on accurate segmentation, which is difficult to achieve on noisy, web-crawled text and may lead to compounding errors. Line-level filtering is inherently ambiguous, as lines in HTML documents represent arbitrary text spans, ranging from single tokens to entire documents. More crucially, granular filtering may cause documents to become less coherent if high-quality text is misclassified and discarded. Since our primary goal is to systematically compare different filtering techniques, we focus on document-level filtering, which provides a clear and unambiguous basis for evaluation while avoiding the complexities of finer-grained methods.

4.1 Related Work

As mentioned above, noisy text in pre-training corpora can degrade the quality of pre-trained LMs. However, there is no clear definition of what precisely constitutes noisy or low-quality text, nor how noisy a document must be to negatively impact downstream performance. For example, documents containing JavaScript code are often considered to be undesirable and specifically targeted for removal using rule-based filters (Raffel et al., 2020). On the other hand, Muennighoff et al. (2023) find that augmenting a monolingual pre-training corpus with Python code, thereby doubling its size, can lead to improved results on downstream tasks. Furthermore, LMs that have been pre-trained on noisy, web-crawled corpora can sometimes outperform comparable models pre-trained on high-quality, curated corpora of similar size, as demonstrated by Artetxe et al. (2022). Despite their noisiness, web-crawled corpora may still be more balanced and representative than curated corpora with regard to downstream datasets. Additionally, some level of noise in the training data can have a regularizing effect during pre-training, which may improve the quality of the model.

Text Quality Corpora

Artetxe et al. (2022) evaluated the quality of documents from EusCrawl and the Basque subsets of mC4 (Xue et al., 2021) and CC100 (Conneau et al., 2020), sampling 100 documents from each. Annotators assessed documents based on five criteria, labeling each as correct or problematic: language correctness (whether the document is in Basque), language variety (whether it is written in standard and correct Basque), coherence, noise, and content (whether the text appears human-generated and meaningful). Each document was then categorized as high, medium, or low quality. Their findings indicate that EusCrawl has, by far, the highest quality among the three corpora, with 66.3% of documents rated as high quality, compared to 31.0% for mC4 and 19.9% for CC100. However, the authors conclude that corpus size and diversity play a more significant role than text quality in downstream performance. A RoBERTa-Base model pre-trained on CC100 achieved an average score of 67.9 on a benchmark of Basque NLP tasks, outperforming models trained on mC4 (67.2) and EusCrawl (66.5). The annotated text quality dataset was not published.

Kreutzer et al. (2022) manually audited the quality of 205 language-specific web-crawled corpora, including 106 parallel corpora and 99 monolingual subsets of OSCAR (Ortiz Suárez et al., 2020) and mC4, both derived from CC. A total of 51 volunteer annotators assessed 100 randomly sampled sentences from each corpus, categorizing them as non-linguistic (NL), in the wrong language (WL), correct but short (CS),

correct but boilerplate or low quality (CB), or correct and natural (CC). Additionally, parallel sentences could be labeled as incorrectly translated (X).

The average sentence length in the monolingual corpora was 28.4 space-delimited tokens, with a median of 19.0. Table 4.1 presents an excerpt from the annotated dataset, illustrating examples of different quality labels.

Language	Text	Label
Galician	Os pecados páganse e o Pontevedra cometeu o máis grave de todos eles, o de non aproveitar as oportunidades que tivo fronte a un Atlético Baleares que se conformaba co empate e atopouse cun "agasallo" inesperado en forma dun gol na única ocasión na que pisou areá granate en toda a segunda parte.	CC
German	Wer heute in die leuchtenden Augen von Sebastian schaut und sein von Herzen kommendes Lächeln sieht, weiß, was Rudy Giovannini ihm geschenkt hat.	CC
Javanese	Episode 102 Subtitle Indonesia4 bulan ago Tonton Now!	WL
Malagasy	#ERROR!	NL
Samoan	ona e le feoti tagata uma i le taimi e tasi, ma ua le afaina lea;	CC

Table 4.1: Five randomly sampled examples from Kreutzer et al. (2022), annotated with quality labels.

Out of the 205 corpora evaluated, the authors found that fewer than 50% of sentences were usable in 87 corpora, while 15 contained no usable data at all. Although there was generally a moderate to strong correlation between resource level and data quality, the authors note that there were several examples of high-resource languages with low-quality data and vice versa. This correlation varied by corpus, being strongest for mC4 (with a Spearman rank correlation of $r = 0.66$) and weakest for OSCAR ($r = 0.37$).

van Noord et al. (2024) annotated text segments from web-crawled corpora in 11 low- and medium-resource languages to assess text quality. For each language, 200 sentences were sampled from language-specific subsets of CC100, mC4, OSCAR, and MaCoCu (Bañón et al., 2022). Unlike the other three corpora, which were derived from CC, MaCoCu was constructed by directly crawling top-level domains associated with these languages.

For each language, two expert annotators evaluated each segment and assigned one of five labels: non-linguistic or in the wrong language (WL), no running text (NR), partially running text (PR), running text with minor quality issues (RT), or publishable text with no quality issues (PT). The annotated text segments contained an average of 31.8 space-delimited tokens, with a median of 19.0, indicating that most examples were approximately the length of a sentence. Table 4.2 presents examples of annotated text.

According to the annotation results, MaCoCu had the highest proportion of high-quality text, with 84.2% of examples containing at least 90% running text. OSCAR

Language	Text	Label
Albanian	Sot mund të jeni protagonist i ndonjë sherri me partnerin, megjithatë përpiquni të mos i çoni gjërat shumë larg! Qiell interesant për përkthyes të pavarur.	RT
Icelandic	Standard-herbergi - 2 meðalstór tvíbreið rúm - gott aðgengi - viðbygging - Baðherbergi	PR
Icelandic	Dagmar - dönsk mynd af slavenska nafninu Dragomír - kær friður. Dragómír hét drottning Valdimars sigursæla Danakonungs (1185-1212) en Danir kölluðu hana Dagmar í merkingunni Dag-mær.	PT
Maltese	Meteogram - 5 gün - Benoni	WL
Turkish	new year's around the world video	WL

Table 4.2: Five randomly sampled examples from van Noord et al. (2024), annotated with quality labels.

followed closely at 84.1%, while CC100 and mC4 had lower proportions at 79.4% and 63.7%, respectively.

To examine the impact of corpus quality on model performance, the authors continued pre-training XLM-R (Conneau et al., 2020) for 50,000 additional steps on each corpus-language combination, training 44 models in total. Despite ranking second-to-last in quality, CC100 achieved the highest average performance, while OSCAR, which ranked among the highest-quality corpora, performed the worst. The authors conclude that corpus size appears to be a stronger indicator of downstream performance than perceived text quality.

Additionally, many datasets exist for tasks related to text quality. To evaluate tools for text extraction and boilerplate removal, Barbaresi (2021) annotated 500 documents in a variety of languages, identifying their main content. For filtering training data for machine translation, Steingrímsson et al. (2023) randomly sampled 10,000 Icelandic sentences from parallel corpora and annotated them as coherent or incoherent. Other types of filtering are also common, such as based on toxicity. Wulczyn et al. (2017) manually annotated 100,000 instances of personal attacks in discussions on the English-language Wikipedia.

Although a variety of text quality datasets have been published, they generally consist of short text segments, typically no longer than a sentence, or focus on only a specific quality issue. To the best of our knowledge, none provide full-document annotations suitable for directly training and evaluating general-purpose document-level text quality classifiers.

Web-Crawled Corpora

Web-crawled corpora are often derived from existing archives of scraped websites. Common Crawl (CC) maintains a massive repository of data crawled from more than 25 billion websites. Many web-crawled corpora are derived from the CC dataset, such as the Colossal Clean Crawled Corpus (C4), which contains 745 GB of English-language text (Raffel et al., 2020). A language classifier was used to identify and extract all

documents within the CC dataset that consist primarily of English-language text. Any document containing the term "lorem ipsum", curly braces, or obscene words were discarded. Lines containing the word "Javascript" or not ending in terminal punctuation marks were removed. Duplicate occurrences of three-line spans were also removed from the corpus. This filtering step reduced the size of the corpus by approximately 88%. The authors found that filtering improved the downstream performance of the T5 model by an average of 1.82 percentage points across an NLP benchmark.

The Multilingual Colossal Clean Crawled Corpus (mC4) is a multilingual version of the C4 corpus, consisting of 6.3 trillion tokens in 101 languages (Xue et al., 2021). The mC4 corpus has been lightly filtered with regard to text quality, using a language classifier to identify the primary language of each document, discarding duplicate occurrences of three-line spans, and removing lines that did not end with a terminal punctuation mark.

MassiveText is an English-language corpus consisting of 2.35 trillion tokens, created for pre-training the Gopher LM (Rae et al., 2022). It is composed of several curated and web-crawled corpora, including MassiveWeb, which contains 506 billion tokens collected using a custom HTML scraper. This corpus was filtered using seven heuristic rules, such as discarding documents if their mean word length falls outside a specified range or if they do not contain a minimum number of unique stop words. The authors found that this filtering results in lower validation loss when pre-training a 1.5 billion parameter version of the Gopher model.

ROOTS is a large, multilingual text corpus, combined from a collection of curated and web-crawled corpora in 46 natural languages (Laurençon et al., 2022). The corpus was filtered using seven document-level heuristic rules, including a maximum perplexity threshold, a maximum word repetition ratio, and a minimum language classification confidence. Fluent speakers for each language determined the thresholds for these rules. ROOTS has been used to pre-train LMs such as BLOOM (Scao et al., 2023).

CulturaX (Nguyen et al., 2023) is a web-crawled corpus that was created by combining multiple web-crawled corpora, all of which are derived from CC. It consists of 6.3 trillion tokens in 167 languages and is filtered using the same rules as ROOTS. For each language, the authors apply a variant of the Interquartile Range (IQR) method (Dekking et al., 2005) by considering the distribution of each metric and setting minimum thresholds at the 10th percentile and maximum thresholds at the 90th percentile. In total, about 39% of the documents were discarded using these settings.

The High Performance Language Technologies (HPLT) project[2] has released large-scale monolingual and bilingual corpora covering 75 languages, totaling 5.6 trillion tokens and 1.85 petabytes in size (de Gibert et al., 2024). The data, sourced from CC and the Internet Archive[3], was processed using language classification, fluency-based filtering, and document-level deduplication. To assess fluency, the authors trained language-specific, word-level 7-gram LMs on up to 200,000 sentences from non-web-crawled sources. More recent releases have expanded significantly, now consisting of 4.5 petabytes of web-crawled text across 193 languages.[4] However, later versions have replaced fluency-based filtering with heuristic rules.

[2]HPLT: `https://hplt-project.org/`

[3]Internet Archive: `https://archive.org/`

[4]HPLT v2.0: `https://hplt-project.org/datasets/v2.0`

Perplexity-Based Filtering

Wenzek et al. (2020) proposed a method for classifying the quality of web-crawled documents based on their perplexity. They trained a 5-gram LM on a high-quality, subword-tokenized corpus and used it to compute perplexity values for documents in a noisy, web-crawled corpus. Applying this approach to English and Polish web-crawled corpora, they divided the documents into low, medium, and high-perplexity segments. By training word embeddings on each segment and evaluating them on semantic and syntactic similarity tasks, they observed that embedding quality degrades as perplexity increases. This method was then used to filter monolingual web-crawled corpora in English, Russian, Chinese, and Urdu. When evaluated on a textual entailment task, LMs pre-trained on the filtered corpora obtained an average accuracy improvement of 3.3 percentage points over models pre-trained on Wikipedia articles.

Muennighoff et al. (2023) found that filtering documents based on perplexity leads to similar or improved downstream performance for pre-trained LMs. In their study, they pre-trained a GPT-2 model with 4.8 billion parameters on a web-crawled English-language corpus. By discarding 75% of the documents with the highest perplexity, they achieved an average improvement of 3.1 percentage points across a benchmark of NLP tasks. Their results suggest that perplexity-based filtering offers greater benefits for noisier corpora and that its effectiveness increases as the model size grows.

Classifier-Based Filtering

Brown et al. (2020) described a logistic regression classifier with bag-of-words representations to distinguish between documents from curated corpora and noisy web-crawled corpora. For each document in the web-crawled corpus, the classifier estimates the probability that it originated from a curated corpus. This approach assumes that a lower probability indicates lower quality. To filter the web-crawled corpus, the authors applied a re-sampling technique based on a probability distribution that heavily favors high-probability documents but still includes a small number of out-of-distribution documents. When applied to a CC dataset of approximately one trillion words, this filtering technique led to around 99% of documents being discarded. The authors did not evaluate the accuracy of the classifier or measure its impact on downstream tasks.

Wu et al. (2021) described a pre-trained LM fine-tuned on a dataset comprising articles labeled as high-quality, low-quality, or advertisements. This model filters out low-quality documents and advertisements from a large web-crawled corpus. After manually reviewing a sample of documents labeled as advertisements (accounting for half of the discarded documents), the authors reported that less than 2% are false positives.

Young et al. (2024) combined heuristic rules, classifiers, and unsupervised semantic clustering to filter a large web-crawled corpus consisting of documents in Chinese and English. The rules discard documents based on length, ratio of special symbols, ratio of short, incomplete, or consecutive sentences, and other metrics. The thresholds for these rules were determined using the IQR method. The classifiers filter documents based on perplexity, quality, coherence, and safety scores. Finally, documents were grouped by semantic similarity, and each cluster was annotated with a quality label. The effectiveness of these filters was not reported.

4.2 Icelandic Text Quality Dataset

In this section, we present TQ-IS, a new text quality dataset for Icelandic, designed to train and evaluate text quality classifiers. It consists of 2,000 documents sampled from three web-crawled corpora: IC3 (see Section 3.2.1), ICC (see Section 3.2.1), and the Icelandic subset of mC4. The source corpora have primarily been filtered using language classifiers and by enforcing a minimum token or character count, but have otherwise undergone minimal filtering with regard to text quality. Each document in TQ-IS contains between 50 and 500 space-delimited tokens. The dataset has undergone a fine-grained analysis identifying and labeling low-quality text spans.[5] Each document has been labeled as "low quality" or "high quality" depending on the proportion of low-quality text it contains, with the two categories being equally represented. We release the TQ-IS dataset with an open license.[6]

4.2.1 Annotation Guidelines

There is no precise definition of what constitutes a high- or low-quality document for pre-training LMs, beyond the impact it may have on the model during training. To create TQ-IS, we focused on documents that were clear examples of either category. A high-quality document primarily consists of running text composed of full, grammatically correct sentences that are connected in a meaningful and coherent way. The text should be properly capitalized and punctuated, with minimal errors. In contrast, low-quality documents are disjointed, incoherent, error-prone, highly repetitive, or largely consist of foreign language text, non-running text, or non-linguistic data.

We classified the following categories of text as low quality:

- **Foreign text**: Text where the primary language is not Icelandic.
- **Non-standard spelling**: Icelandic text that does not conform to modern standards of spelling or grammar.
- **Corrupted text**: Icelandic text that contains character encoding errors (e.g., "Reykjav??k"), HTML character entities (e.g., """), soft hyphens, and escaped characters (e.g., "\n" and "\u266c").
- **Run-on text**: Icelandic text that contains a large number of run-on sentences or words.[7]
- **OCR text**: Digitized Icelandic text that contains a large number of errors and flaws caused by the optical character recognition (OCR) process (e.g., misrecognized characters or text columns appearing out of order).
- **Non-linguistic text**: Text with no apparent meaning (e.g., seemingly random sequences of symbols and numbers).
- **Incoherent text**: An apparently meaningless sequence of Icelandic words.
- **Code**: Text that consists primarily of code, such as HTML or JavaScript.

[5] This work was carried out solely by the author of this thesis.

[6] `https://github.com/jonfd/tq-is`

[7] A run-on sentence occurs when two independent clauses run together without proper punctuation or appropriate conjunctions.

- **Non-content text**: Icelandic text that does not contribute to the main subject of the document (e.g., boilerplate text, headers, footers, metadata, and navigational elements).
- **Non-running text**: Icelandic text that is relevant to the main subject of the document but is not in the form of full, grammatically structured sentences or breaks the flow of the document (e.g., lists, bullet points, tabulated data, and image captions).
- **Fragmented text**: Icelandic text that lacks flow or continuity (e.g., a list of headlines from news article or a sequence of short, truncated previews from unrelated blog posts).
- **Low-quality translations**: Text that has been poorly translated into Icelandic.
- **Repetitive text**: Icelandic text that has occurred elsewhere in the document.

These categories are intended to identify text that could potentially degrade the quality of monolingual pre-trained LMs. Due to the fine-grained labels, specific categories may be excluded for tasks where the definition of low-quality text might differ.

We manually identified low-quality text spans in each document and labeled them according to the categories listed above. Documents were classified as low quality if at least one-third of the text consisted of low-quality spans, and as high quality if 10% or less was low-quality text. Our analysis revealed that documents in TQ-IS are less ambiguous than initially suspected. We found that 93% of the high-quality documents contain no low-quality text whatsoever, while 80% of low-quality documents consist of text where at least 90% is of low quality.

4.3 Rule-Based Filtering

In this section, we give an overview of 13 different heuristic rules that can be used to filter web-crawled corpora and evaluate them on the TQ-IS dataset. For the best-performing ruleset, we compare the results obtained using the optimal threshold values to those obtained using the IQR method.

4.3.1 Rules

We describe 12 document-level rules that were used to filter the ROOTS and MassiveWeb corpora and propose one additional rule based on our analysis of low-quality documents in the TQ-IS dataset.

4.3.1.1 ROOTS

In our experiments, we evaluated several rules used to filter the ROOTS corpus. We omitted one rule that discards documents containing too many sexually explicit words, as such word lists are not readily available for all languages. We also excluded a rule that discards documents based on word count, as documents in TQ-IS are already limited to between 50 and 500 space-delimited tokens.

Perplexity A LM is trained on a high-quality corpus. This model is then used to calculate the perplexity score of a document, estimating how likely it is that the model could generate the same text. Higher perplexity indicates less predictable text. Therefore, it is assumed that low-quality documents will be difficult to predict and will have a high perplexity value. Consequently, documents with a perplexity score above a certain threshold are considered to be of low quality and are discarded.

Character Repetition Ratio This rule targets documents with a high proportion of repeated character n-grams, indicative of automatically generated text or text-based visuals (e.g., log files or ASCII art). This ratio is calculated as the number of frequently occurring character n-grams divided by the total number of character n-grams. Documents with a character repetition ratio exceeding a maximum threshold are discarded.

Word Repetition Ratio Similarly, the word repetition ratio of a document is calculated by dividing the number of frequently repeated word n-grams by the total number of word n-grams. A high ratio may suggest that a document contains a large amount of automatically generated text, spam, or content intended for search engine optimization (e.g., keywords that are repeated to increase search rankings). Documents with a high word repetition ratio are discarded.

Special Character Ratio Documents with a large proportion of non-alphabetic characters, such as emojis, symbols, digits, and punctuation marks, may be corrupted (e.g., due to incorrect character encoding) or otherwise contain a limited amount of natural language text. Documents with a special character ratio exceeding a maximum threshold are discarded.

Stop Word Ratio In the context of text quality filtering, stop words generally consist of common function words, i.e., words that serve a syntactically and grammatically important purpose but lack significant meaning on their own. This generally includes word classes such as conjunctions, prepositions, pronouns, and articles. Documents with a very low ratio of stop words are unlikely to contain coherent text in a natural language, and are discarded.

Language Confidence Score A language classifier is used to determine the primary language of each document. If the primary language is not targeted for inclusion in the corpus or if the confidence falls below a certain threshold, the document is discarded.

4.3.1.2 MassiveWeb

We also consider the rules that were used to filter the MassiveWeb corpus, omitting a rule that enforces a minimum and maximum word count for documents.

Mean Word Length Documents with a mean word length falling outside the expected range may lack natural language text or be malformed (e.g., containing poorly digitized text where spaces have been frequently inserted or removed). Only documents with a mean word length within the specified range are retained.

Symbol to Word Ratio A high ratio of hashtags or ellipses to words may suggest that documents largely consist of keywords or truncated text. If this ratio exceeds a maximum threshold, the document is discarded.

Initial Bullet Point Ratio Documents with a high proportion of lines starting with bullet points are likely to consist of itemized lists rather than running text. If this ratio exceeds a maximum threshold, the document is discarded.

Trailing Ellipsis Ratio A high proportion of lines ending with an ellipsis may suggest that a document contains a large amount of truncated text. Documents for which this ratio exceeds a maximum threshold are discarded.

Alphabetic Character Ratio A low ratio of tokens containing at least one alphabetic character within a document may suggest that the text is primarily non-linguistic. If the ratio falls below a minimum threshold, the document is discarded.

Unique Stop Word Count If a document does not contain at least two unique stop words, it is discarded.

4.3.1.3 Other Rules

We propose one additional rule based on observations from the TQ-IS dataset.

Mean Subword Length Subword tokenizers break out-of-vocabulary words into sequences of known subwords (Wu et al., 2016). As a result, documents containing a large amount of non-linguistic text, foreign words, numbers, URLs, or other uncommon tokens tend to have shorter subwords on average. We propose a new rule for discarding documents with a mean subword length (i.e., the average number of characters per subword) below a minimum threshold.

4.3.2 Experimental Setup

In this section, we describe how we performed feature extraction and selected the optimal ruleset and thresholds.

4.3.2.1 Feature Extraction

Most features were extracted in a straightforward manner, though some require additional explanation. For perplexity, we followed the method described by Wenzek et al. (2020), where a curated, subword-tokenized corpus was used to train an n-gram LM with the KenLM toolkit (Heafield, 2011). In their work, a Unigram tokenizer with a vocabulary size of 65,536 and a 5-gram LM was used. We searched for the optimal combination of vocabulary size and n-gram order that maximized the F_1 score of a perplexity-based classifier on the TQ-IS dataset. Specifically, we evaluated WordPiece tokenizers with vocabulary sizes of 8k, 16k, and 32k and n-gram orders of 2, 3, and 4. Using the IGC (see Section 3.2.1) as our high-quality corpus, we evaluated each model

using stratified 10-fold cross-validation[8] with 10 repetitions on the TQ-IS dataset and selected the configuration that obtained the highest F_1 validation score.

Character and word repetition ratios were calculated based on the proportion of recurring n-grams. We evaluated character n-gram orders from 2 to 20 and word n-gram orders from 2 to 10, selecting the order that achieved the highest F_1 score on the TQ-IS dataset when used in conjunction with other rules. In both cases, we found that the choice of n-gram order has limited impact, as different orders generally yield the same F_1 score, though at slightly adjusted thresholds. Based on these findings, we calculated 5-gram word and 10-gram character repetition ratios for these features.

For language confidence scores, we used the *langid.py* library (Lui and Baldwin, 2012) to assign a confidence score to each document in TQ-IS. For documents for which Icelandic is not the primary language, we set the confidence score to zero.

4.3.2.2 Threshold Optimization

Given the large search space for the full ruleset, we used *forward feature selection* to identify the optimal set of rules and threshold values. Starting with an empty ruleset, we added the rule that yielded the highest F_1 score on the TQ-IS dataset. We then iteratively added the rule that provided the greatest improvement to the F_1 score when combined with the current ruleset, until no further improvement was possible or all available rules had been selected. For each rule, we evaluated a range of threshold values, beginning just before the point where the first false negative occurred (i.e., a high-quality document misclassified as low-quality) and extending until an F_1 score of 95% could no longer be achieved.

4.3.3 Results

In this section, we present the results of our experiments aimed at identifying the optimal configuration for calculating perplexity, determining the best ruleset and threshold values for filtering the TQ-IS dataset, and evaluating the effectiveness of the IQR method for selecting thresholds in heuristic rule-based filtering.

4.3.3.1 Perplexity

To identify the optimal n-gram model configuration for text quality classification, we evaluated each n-gram model on the TQ-IS dataset using stratified 10-fold cross-validation with 10 repetitions. A bigram model with a vocabulary size of 32k achieved the highest average F_1 validation score of 94.60%, outperforming all other models by a statistically significant margin. The average F_1 validation score for each model is shown in Table 4.3.

We observed that lower-order n-gram models with higher vocabulary sizes are generally more effective for perplexity-based filtering. Although higher-order n-gram models are more expressive, they often produce lower perplexity values for foreign-language documents, which account for approximately 40% of the low-quality documents in the TQ-IS dataset. As a result, they are more likely to misclassify foreign-language text as high quality.

[8] In our case, a stratified cross-validation means that we ensured that each fold contained approximately same ratio of low-quality and high-quality documents.

Vocab.	2-gram	3-gram	4-gram
8k	93.29%	93.89%	92.88%
16k	93.37%	93.50%	92.48%
32k	**94.60%**	93.89%	93.15%

Table 4.3: Average F_1 validation scores for each n-gram model on the TQ-IS dataset. The best score is shown in bold; all others are statistically significantly different (paired t-test with Holm-Bonferroni correction; $p < 0.05$).

In the IGC, foreign-language words are relatively rare and are typically broken down into sequences of very short subwords, even when using large vocabularies. Although such sequences are uncommon, they are often highly predictable, resulting in lower perplexity values. Higher-order n-gram models, which can capture more of these rare but predictable subword sequences, thus become less effective at distinguishing between different languages. In contrast, lower-order models, which are constrained by a shorter context window, are less susceptible to this issue.

Additionally, our findings indicate that larger vocabulary sizes improve an n-gram model's ability to identify incoherent or disjointed text by providing greater context between adjacent words. Smaller vocabularies, on the other hand, result in words being broken into longer sequences of shorter subwords, reducing the average context window. For example, a bigram model with a small vocabulary size might only capture the very end of one word and the beginning of the next. If the text is incoherent and those two words are not connected in a meaningful way, the limited context window might prevent this incoherence from being reflected in the document's perplexity value.

For the best-performing model, the majority of classification errors involve low-quality documents containing a large amount of non-running text, and high-quality documents classified as low quality. In the case of non-running text, most errors occur in documents containing long lists of numbers or proper names, which typically have low perplexity values. High-quality documents that were misclassified often contain proper names in foreign languages, though they rarely appear to be the primary cause of the misclassifications. Beyond this, we can see no significant or consistent patterns that contribute to classification errors.

4.3.3.2 Rule-Based Approach

When performing forward feature selection on the TQ-IS dataset, our results show that perplexity is the single most effective feature for distinguishing between low and high-quality documents. Evaluated individually, we found the optimal maximum perplexity threshold to be 395.7, yielding an average F_1 score of 94.60%. We observed that the optimal perplexity threshold is relaxed significantly as more rules are added, rising to 492.0 for the optimal ruleset. We found that the optimal F_1 score is obtained by applying a combination of six rules, leaving seven rules unused. The rules and their overall impact are shown in Table 4.4.

When we visualize the distribution of documents in TQ-IS by feature pairs, as shown in Figure 4.1, we observe that high-quality documents form a single, dense cluster, while low-quality documents appear as outliers. Overlaying the optimal thresholds on this plot reveals that they align precisely at the boundary that separates the dense cluster from the outliers. An examination of the distribution of documents in the

Metric	Ratio	F_1 score
Perplexity	43.90%	94.60%
+ Stop word ratio	32.40%	97.33%
+ Mean subword length	41.00%	97.82%
+ Word repetition ratio	4.20%	98.11%
+ Character repetition ratio	7.00%	98.21%
+ Mean word length	5.15%	98.26%

Table 4.4: Optimal ruleset and thresholds obtained for the TQ-IS dataset using cross-validated forward feature selection. The rules are listed in order of selection. The table shows the F_1 score of each rule when applied in conjunction with the rules above it, and the ratio of documents that fall outside the optimal threshold for each rule. In total, 49.65% of the documents are filtered using these rules.

Icelandic subset of the mC4 corpus reveals similar properties, with the optimal thresholds for the TQ-IS dataset neatly separating the dense cluster from its outliers. This suggests that a visual inspection of the distribution of documents may suffice to estimate near-optimal threshold values for heuristic rules, without requiring a manually annotated dataset.

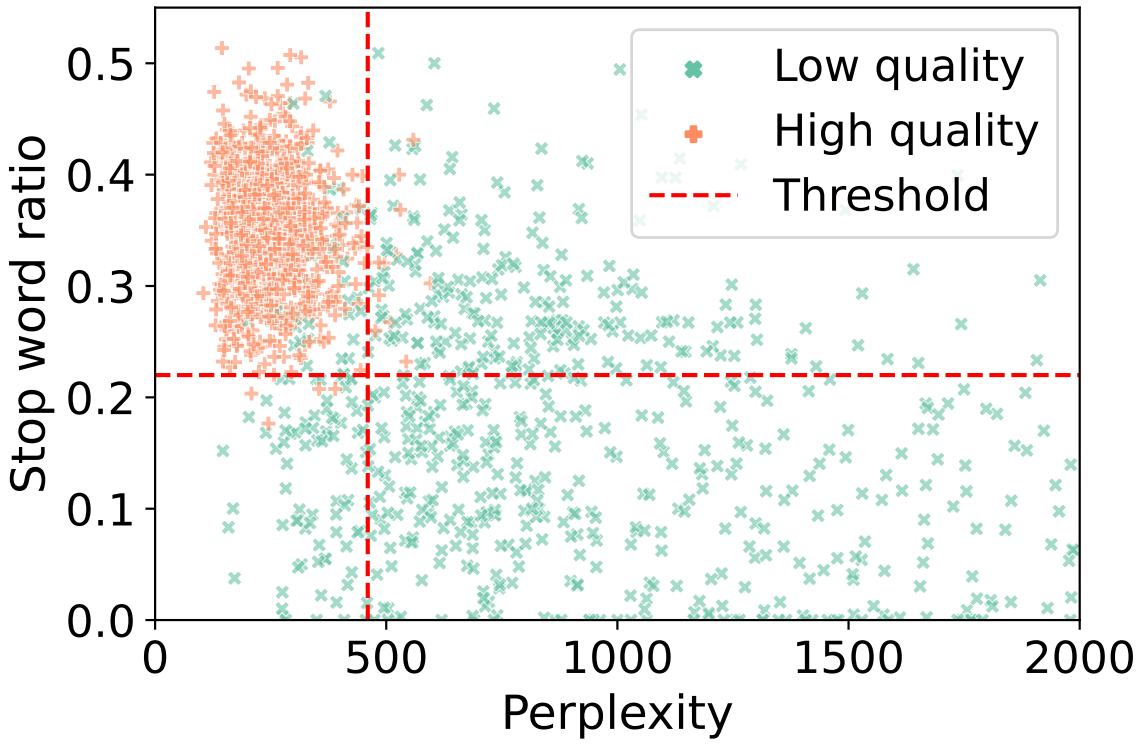

Figure 4.1: Distribution of documents in the TQ-IS dataset based on their perplexity score and stop word ratio. The red, dashed line shows the optimal perplexity and stop word ratio thresholds found using forward feature selection.

Our findings on text quality filtering for Icelandic can be contextualized alongside recent large-scale filtering efforts for English web data, such as the FineWeb study (Penedo et al., 2024). The authors of FineWeb applied numerous heuristic rules from the MassiveWeb and C4 rulesets (see Sections 4.3.1.2 and 4.1), along with additional

rules of their own design, to improve the quality of a massive English-language corpus derived from CC. While both studies focus on rule-based filtering, our experiment involved finding an optimal ruleset and corresponding threshold values using a manually annotated dataset, whereas FineWeb evaluated the effectiveness of heuristic rules based on their impact on downstream performance, applying fixed thresholds.

FineWeb evaluated several rules individually and found that most provided modest improvements, with larger rulesets yielding more significant benefits. In contrast, our results showed that the choice of threshold values is crucial and requires optimization, that optimal thresholds differ depending on the ruleset, and that rules which are effective in isolation may become redundant when combined with others. Notably, the three most effective rules in our study (perplexity, stop word ratio, and mean subword length) were not included in FineWeb's filtering pipeline. While FineWeb already demonstrates significant benefits from rule-based filtering, our findings suggest that refining rule selection and optimizing threshold values could further improve the effectiveness of their filtering approach, potentially allowing for more efficient filtering with fewer, more targeted rules.

4.3.3.3 Interquartile Range

We also evaluated the IQR method for selecting minimum and maximum thresholds, as described by Nguyen et al. (2023) (see Section 4.1). In this approach, all thresholds are configured to discard the same proportion of documents. For example, we might set the maximum thresholds (e.g., for perplexity and word repetition ratio) to the 90th percentile, and minimum thresholds (e.g., for stop word ratio and mean word length) to the 10th percentile. For the six rules shown in Table 4.4, this results in an F_1 score of 80.75%, which is far below the score obtained with the optimal thresholds.

Under optimal settings, between 4.2% and 43.9% of documents fall outside the acceptable range for each rule. Setting a uniform threshold somewhere in between leads to poor overall results. Having each rule discard the same proportion of documents results in some rules being underutilized (e.g., perplexity and mean subword length) and others being applied much too aggressively (e.g., word repetition ratio). Therefore, we conclude that the IQR method is not an ideal approach for approximating optimal thresholds for text quality filtering.

4.4 Classifier-Based Filtering

In this section, we evaluate several classifier-based approaches for text quality filtering using the TQ-IS dataset. We consider four types of classifiers: a perplexity-based classifier, a supervised classifier trained on labeled data, a weakly supervised classifier, trained to discern between documents from curated and web-crawled corpora, and a novel, unsupervised classifier based on clustering and outlier detection algorithms.

4.4.1 Classifiers

We evaluate each approach in detail, describing how documents are classified as either low or high quality.

4.4.1.1 Perplexity-Based Classifier

As described in Section 4.3.2.1, a typical perplexity-based filtering approach involves calculating the perplexity of documents in a web-crawled corpus using a LM trained on a high-quality corpus. If the perplexity of a document exceeds a predetermined threshold, the document is considered to be of low quality and is discarded.

This method could be considered a hybrid approach. While a trained model is used to extract the perplexity value of a document, the classification itself is performed using a predetermined threshold, which is more common in rule-based approaches. In the literature, this approach has been categorized both as rule-based (Laurençon et al., 2022; Öhman et al., 2023) and as classifier-based (Young et al., 2024). For this reason, we include perplexity-based filtering in our evaluation. We used the same feature extraction approach as described in Section 4.3.2.1. Following the results obtained in Section 4.3.3.1, we calculated perplexity using a bigram LM and a subword tokenizer with a vocabulary size of 32k.

4.4.1.2 Supervised Classifier

We trained a supervised classifier to distinguish between low and high-quality documents in the TQ-IS dataset, following the approach of Wu et al. (2021). Our method involves fine-tuning a pre-trained LM on the TQ-IS dataset. As the LM has a limited context length, we generate training samples by applying a sliding window across each document. The classifier is trained to assign quality labels (low or high) to these individual windows. The document-level classification is subsequently determined by evaluating the proportion of high-quality windows within a document. A document is classified as high quality if this proportion exceeds a threshold determined from the training data; otherwise, it is labeled as low quality.

4.4.1.3 Weakly Supervised Classifier

We employed a weakly supervised approach inspired by Brown et al. (2020), training a classifier to distinguish between documents from a curated, high-quality corpus and those from a noisy, web-crawled corpus. This approach falls under *inaccurate supervision*, a type of weakly supervised learning where training labels are not always ground truth for the target task (Zhou, 2017). The classifier assigns scores to unseen documents based on the probability that they originate from the high-quality corpus. Although this approach eliminates the need for a manually labeled training dataset, it may prove challenging to accurately translate this probability into the appropriate label without access to such a dataset. To enable a direct comparison to the supervised classifier, we used the same approach as before, fine-tuning a pre-trained LM on the dataset and generating training samples using a sliding window.

4.4.1.4 Unsupervised Classifiers

A scatter plot of feature pairs from the TQ-IS dataset, shown in Figure 4.1, reveals that high-quality documents form a dense, well-defined cluster, while low-quality documents are more sparsely distributed. This suggests that it may be possible to accurately classify documents as low or high quality using unsupervised clustering or outlier detection algorithms. We evaluated three such algorithms:

Gaussian Mixture Model
A Gaussian Mixture Model (GMM) is a probabilistic approach used to model a dataset as a combination of several Gaussian distributions. Each distribution is characterized by parameters such as means, covariances, and mixture weights. It can be applied as a clustering algorithm under the assumption that each Gaussian distribution corresponds to a distinct cluster. It provides a soft clustering approach, meaning it can be fitted to one dataset and then used to probabilistically assign each data point in another dataset to the resulting clusters.

One-Class Support Vector Machine
A One-Class Support Vector Machine (OCSVM) (Schölkopf et al., 2001) is an outlier detection algorithm that maps data into a high-dimensional feature space using a kernel function. Then, it finds the smallest possible boundary that encloses the densest region of the data while maximizing the distance between the boundary and the origin of the feature space. Data points that fall outside this boundary are considered outliers.

Isolation Forest
An Isolation Forest (Liu et al., 2008) is an outlier detection algorithm that generates a forest of binary trees by recursively and randomly splitting the dataset until all data points have been isolated. Each data point is scored based on the average number of splits required to isolate it. Data points with lower average scores are more likely to be outliers under the assumption that outliers are few and different, and thus easier to isolate.

4.4.2 Experimental Setup

In this section, we describe the configuration and evaluation process for each classifier.

4.4.2.1 Perplexity-Based Classifier

We used the same implementation as described in Section 4.4.1.1, where a bigram LM is trained on a high-quality corpus that has been processed using a subword tokenizer with a vocabulary size of 32k. As in Section 4.3.3.1, we evaluate the classifier using stratified 10-fold cross-validation, determining the optimal threshold on the training set and evaluating it on the validation set.

4.4.2.2 Supervised Classifier

We trained the supervised classifier by pre-training a TEAMS-Small model (Shen et al., 2021) on the IGC and fine-tuning it on the TQ-IS dataset. The classifier was evaluated with stratified 10-fold cross-validation. We generated 2,401 sliding windows of 512 tokens with a stride of 256 tokens. Each window was labeled as low quality if over one-third of its text consists of low-quality spans; otherwise, it was labeled as high quality. This resulted in 1,342 low-quality windows and 1,059 high-quality windows.

For document-level classification, the quality of all windows within a document was predicted, and the ratio of high-quality windows was computed. Documents were classified as high quality if this ratio exceeded a predetermined threshold. The model was fine-tuned on the training set, which was also used to select the threshold value

that maximizes the F_1 score of the classifier. This threshold was then evaluated on the validation set.

4.4.2.3 Weakly Supervised Classifier

Building on the approach used for the supervised classifier, we fine-tuned a TEAMS-Small LM on a balanced dataset of 50,000 high-quality documents sampled from the IGC and 50,000 web-crawled documents from the Icelandic subset of the mC4 corpus. Training samples were generated using the same sliding window approach, with a window size of 512 tokens and a stride of 256 tokens, and each window was labeled according to its source corpus.

To classify windows as low or high quality, we predicted the probability of each window originating from the high-quality corpus. If this probability exceeded a certain threshold, the window was classified as high quality. We evaluated the weakly supervised classifier with stratified 10-fold cross-validation, using the training set to select thresholds for both window-level and document-level classification, optimizing for F_1 score.

4.4.2.4 Unsupervised Classifiers

For the unsupervised algorithms, we used the same features that were used for the rule-based approach (e.g., perplexity, character repetition ratio, word repetition ratio, etc.). We optimized the parameters of each algorithm to maximize the F_1 score on the TQ-IS dataset. We used the *scikit-learn* library for Python (Pedregosa et al., 2011) to implement the three clustering and outlier detection algorithms.

As OCSVM is sensitive to extreme outliers, we scaled the features using scikit-learn's robust scaler. For GMM, we found that trimming the training set by removing documents with a perplexity value of 3,100 or higher yields better results than using the robust scaler. Our experiments show that GMM models perform best when trained on a noisy, web-crawled corpus, while OCSVM and Isolation Forest models yield better results when trained on a high-quality corpus.

To optimize the parameters, we fit a GMM model to the Icelandic subset of the mC4 corpus, while fitting the OCSVM and Isolation Forest models to the IGC. We trained each model on a sample of 50,000 documents, as we found that larger training sets did not lead to improved performance. We then created a stratified 10-fold split of TQ-IS, using 90% of the documents for validation and 10% for testing in each fold. We selected the parameters that obtained the highest average F_1 score on the training sets, and report the average results obtained on the validation sets.

4.4.3 Results

We evaluated each of the classifiers on the TQ-IS dataset. An overview of the results can be seen in Table 4.5.

4.4.3.1 Perplexity-Based Classifier

As shown in Section 4.3.3.1, the perplexity-based classifier obtains an average F_1 test score of 94.60% when evaluated on the TQ-IS dataset.

Classifier	F_1 score
Supervised classifier	99.01%
Gaussian Mixture Model	98.32%
Isolation Forest	97.52%
One-Class SVM	96.40%
Perplexity-based classifier	94.60%
Weakly supervised classifier	93.34%

Table 4.5: F_1 scores on the TQ-IS dataset for the classifiers. The GMM and Isolation Forest models obtained the best results using perplexity, stop word ratio and mean subword length as features, while OCSVM performed best using only perplexity and stop word ratio.

4.4.3.2 Supervised Classifier

For window-level classification, we found that the supervised classifier obtained an average F_1 test score of 96.80% when evaluated on the TQ-IS dataset using stratified 10-fold cross-validation. For each fold, we then computed the ratio of high-quality windows within each document in the training set. We found that a threshold of 66.67% maximized the document-level F_1 score on the training set, meaning that at least two thirds of the windows within a document must be classified as high quality for the document itself to also be classified as high quality. This is consistent with our annotation guidelines, according to which a document is considered low quality if at least one third of its contents are of low quality. Using this threshold, the classifier obtains an average F_1 test score of 99.01%.

4.4.3.3 Weakly Supervised Classifier

We fine-tuned a TEAMS-Small model on a sample of 50,000 documents from the IGC and 50,000 documents from the Icelandic subset of the mC4 corpus. For each window in the TQ-IS dataset, we used this model to predict the probability of the window text originating from the IGC. We performed stratified 10-fold cross-validation on the TQ-IS dataset, determined the threshold values that yielded the highest F_1 score on the training set, and evaluated them on the test set.

We found that the optimal threshold for window-level classification was extremely low, with a value of 0.00113, which results in an average window-level F_1 test score of 88.98%. We found that the optimal threshold for classifying documents based on the ratio of high to low-quality windows is 50%, meaning that at least half of the windows have to be classified as high quality for the document to be labeled as high quality as well. This results in an average document-level F_1 test score of 94.34%.

Unlike the perplexity-based classifier, the weakly supervised classifier proves more capable when it comes to detecting documents with fragmented text. However, it also struggles with non-running text. This may be due to the fact that non-running text is represented to some degree in IGC. Additionally, we find that it does not perform as well as the perplexity-based classifier when it comes to identifying documents that contain a large number of OCR errors.

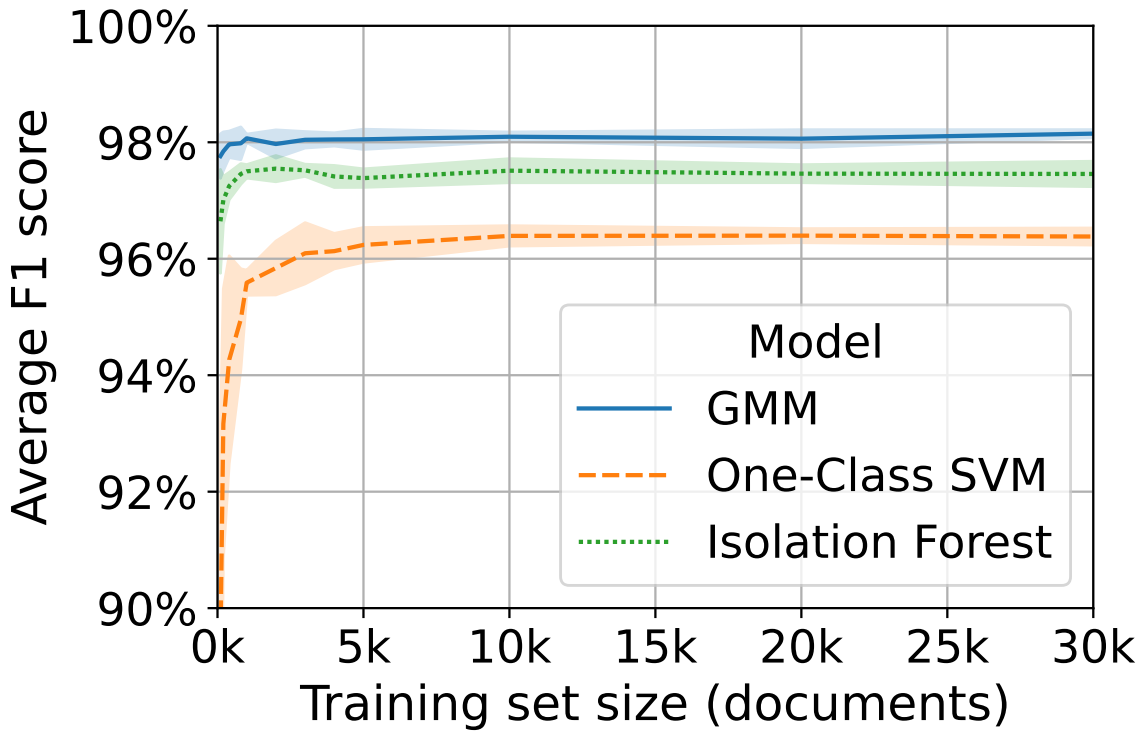

Figure 4.2: Average F_1 scores obtained by the three clustering and outlier detection algorithms on TQ-IS. The results show that a GMM performs very well even when fitted only to a handful of web-crawled documents, and that OCSVM and Isolation Forest models only require a small number of high-quality documents to be able to effectively identify low-quality outliers.

4.4.3.4 Outlier Detection

We observed that the optimal set of features for all three clustering and outlier detection algorithms is smaller than the number of metrics used for the optimal rule-based approach, with OCSVM using only two features. This may be explained, in part, by the fact that the modest benefit to the F_1 score offered by some rules, such as word repetition ratio (+0.29%) and special character ratio (+0.05%) (see Table 4.4), may not make up for the cost of increasing the dimensionality of the data by adding a new feature.

To determine the impact of the training set size on the performance of the three models, we evaluated them on a variety of training set sizes, ranging from 100 to 30,000 documents. For each size, we sampled ten distinct training sets from the appropriate corpus (mC4 for GMM and IGC for OCSVM and Isolation Forests) and report the average F_1 score obtained on TQ-IS.

As Figure 4.2 shows, we observe significantly diminished returns for all three methods after increasing the training set size to around 5,000 documents. Notably, the GMM model appears to be the most robust of the three, maintaining the most stable score and exhibiting the smallest standard deviation. These results indicate that the methods are likely to be effective even for under-resourced languages where web-crawled text may be limited.

4.5 Text Quality and Downstream Performance

In this section, we evaluate the impact of text quality filtering on the downstream performance of pre-trained LMs for low and medium-resource languages. Previously, we demonstrated that both rule-based and classifier-based filtering methods achieved high F_1 scores on the TQ-IS dataset, ranging from 93.40% to 99.01%. These results indicate strong agreement across methods. Based on these results, we selected a single filtering approach for evaluating downstream performance.

Although the supervised classifier obtained the highest F_1 score on the TQ-IS dataset, it requires manually labeled training data, which we lack for languages other than Icelandic. In contrast, both rule-based and GMM-based classifiers can be tuned through visual inspection of the corpus, while obtaining only slightly lower scores. For the rule-based method, thresholds can be set at the boundaries of the high-quality cluster (see Figure 4.1), while the parameters of the GMM classifier can be iteratively adjusted until it accurately captures the high-quality cluster. We selected the GMM-based approach as the single filtering approach, as it performed similarly to or better than the rule-based method while using fewer features.

In the TQ-IS dataset, high-quality documents form a dense, elliptical cluster characterized by low perplexity, high stop word ratios, and high mean subword lengths (see Figure 4.1). Conversely, low-quality documents are sparsely distributed and appear as outliers. These properties align well with the strengths of the GMM classifier, which achieved an F_1 score of 98.32% on the TQ-IS dataset. We observed similar clustering patterns across multiple languages in the mC4 corpus, suggesting that the GMM approach is well-suited for filtering noisy, web-crawled corpora.

To evaluate the impact of text quality filtering, we trained and applied GMM-based classifiers to the Icelandic, Estonian, Basque, Galician, Nepali, and Tajik subsets of the mC4 corpus. For each language, we pre-trained TEAMS-Small LMs on both filtered and unfiltered corpora and compared their downstream performance across a benchmark of NLP tasks. Table 4.6 provides an overview of the tasks and datasets in the evaluation benchmark.

For Icelandic, we conducted additional experiments to investigate how model size affects sensitivity to noise. We pre-trained both TEAMS-Small and TEAMS-Base models to determine whether larger models are more robust to noise, such as by better distinguishing between signal and noise during pre-training, or more sensitive, potentially overfitting to random noise and patterns in low-quality text. Furthermore, we evaluated the impact of augmenting a high-quality corpus with web-crawled text, both filtered and unfiltered, representing a more realistic scenario for languages where curated corpora are available.

4.5.1 Experimental Setup

In this section, we describe how we trained GMM classifiers for each language and used them to filter the mC4 corpus, as well as how we pre-trained LMs on the filtered and unfiltered corpora.

4.5.1.1 Filtering

For each language, we trained a GMM classifier using the three features which yielded optimal F_1 scores on the TQ-IS dataset: perplexity, stop word ratio, and mean subword

Language	Task	Dataset	Size
Icelandic	POS	MIM-GOLD	1,000,218
Icelandic	NER	MIM-GOLD-NER	1,000,231
Icelandic	DP	UD Icelandic IcePaHC	983,668
Icelandic	ATS	IceSum	1,000
Icelandic	QA	RUQuAD	10,530
Estonian	POS	UD Estonian EDT	440,000
Estonian	NER	EstNER	184,638
Estonian	DP	UD Estonian EDT	440,000
Basque	POS	UD Basque BDT	121,443
Basque	NER	EIEC	59,759
Basque	DP	UD Basque BDT	121,443
Galician	POS	CTG	1,196,734
Galician	NER	NERC	202,334
Galician	DP	UD Galician TreeGal	23,479
Nepali	POS	NNC-CS	1,123,528
Nepali	NER	EverestNER	308,353
Nepali	TC	SIB-200	1,004
Tajik	NER	WikiANN	300
Tajik	TC	SIB-200	1,004

Table 4.6: Languages, tasks and datasets in the evaluation benchmark, including dataset size measured in number of labeled examples.

length. Perplexity was computed using a bigram model trained on a subword-tokenized high-quality corpus, as detailed in Section 4.3.3.1.

The GMM classifier requires two key parameters to be configured: the number of mixture components (i.e., clusters) and the perplexity threshold for trimming the training dataset. These were adjusted iteratively based on visual inspection of the resulting clusters:

Number of Components We initialized the classifier with five components and adjusted this value iteratively. The goal was to ensure that high-quality documents formed a single cluster. If high-quality documents were split across multiple clusters, we reduced the number of components. Conversely, if the high-quality cluster was too large (i.e., contained many outliers), we increased the number of components.

Perplexity Threshold We used an initial threshold of 3,000, which we adjusted to control the size of the high-quality cluster. Increasing the thresholds expanded the cluster, while decreasing it reduced its size. Significant changes to the threshold value may affect the number of clusters that are present in the dataset, which may require adjusting the number of components.

4.5.1.2 Pre-Training

We pre-trained TEAMS-Small models on the filtered and unfiltered corpora for each language and evaluated them on a benchmark of downstream tasks. A WordPiece tokenizer with a vocabulary size of 32k was trained separately for each corpus.

During pre-processing of the unfiltered Nepali corpus, we identified a significant number of Chinese-language documents. To address this, spaces were inserted around all Chinese characters before training the tokenizer, ensuring that the vocabulary consisted primarily of Nepali subwords. As the vocabulary was restricted to containing a maximum of 1,000 single-character subwords, the proportion of Chinese-language subwords in the final vocabulary was substantially reduced.

4.5.2 Filtering

The predictions of the GMM classifier for each language are visualized in Figure 4.3. Across all corpora, we observed a dense cluster of documents characterized by low perplexity, high mean subword length, and high stop word ratio, consistent with high-quality text. In each case, the GMM classifier effectively isolated documents within this cluster. Although we lack manually labeled text quality datasets for languages other than Icelandic, these visualizations strongly support the hypothesis that clustering and outlier detection algorithms are effective for text quality filtering across diverse languages.

Further analysis revealed additional dense clusters of low-quality documents in the Basque, Galician, Nepali, and Tajik subsets:

Basque A small, dense cluster of approximately 26,000 documents, characterized by unusually low perplexity, stop word ratio, and mean subword length. These documents primarily consisted of automatically generated Wikipedia articles, created from the same template, describing cities and geographic areas. Additionally, the bigram model used for calculating document perplexity was trained on the EusCrawl corpus, which contains 66 million tokens from Wikipedia. As the text is highly predictable, and since there may be some overlap with EusCrawl, each document has a very low perplexity value. The low stop word ratio and mean subword length were due to the prevalence of numbers and foreign place names. However, this cluster constituted only a small fraction of the Basque corpus.

Galician Approximately 320,000 documents formed a distinct cluster with low perplexity, stop word ratio, and mean subword length. Most of these documents originated from an online book database, containing information such as book titles, author names, and ISBN identifiers. However, they also included a substantial amount of malformed JavaScript code.

Nepali A large cluster of around one million documents, accounting for over a third of the corpus, characterized by extremely low stop word ratio and mean subword length. Analysis revealed that these documents were predominantly in Chinese, suggesting significant language classification issues within the mC4 corpus.

Tajik Language classification errors were particularly pronounced in the Tajik corpus, where approximately 700,000 documents, or 55% of the corpus, formed a cluster with high perplexity and low stop word ratio and mean subword length. These documents were primarily in Uzbek but misclassified as Tajik, despite having a different alphabet, belonging to a different language family, and not being mutually intelligible.

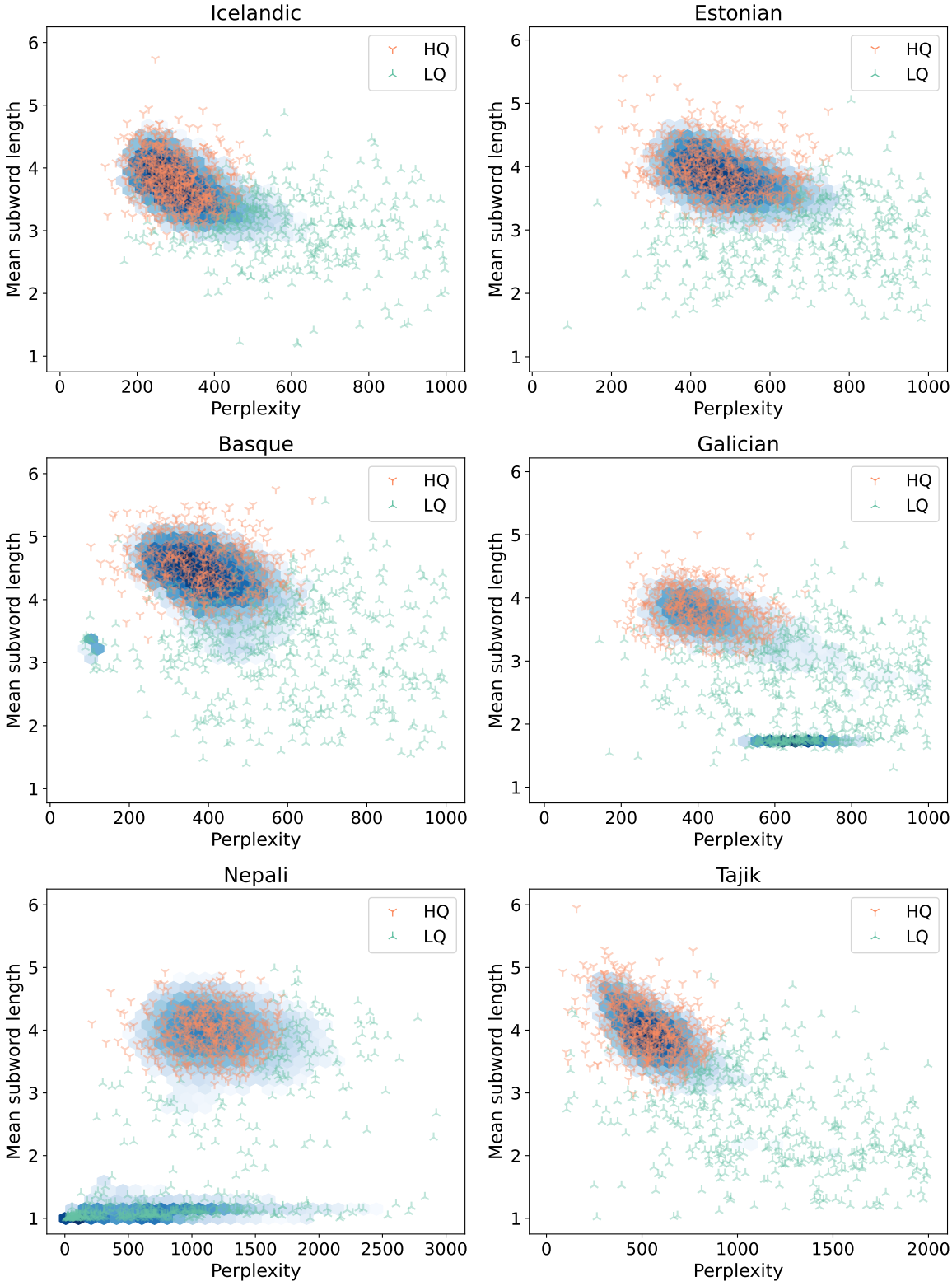

Figure 4.3: GMM classifier predictions for the Icelandic, Estonian, Basque, Galician, Nepali, and Tajik subsets of the mC4 corpus. Scatter plots depict 1,000 predictions for each language, overlaid on a hexbin plot showing document distributions based on perplexity and mean subword length. The low-quality cluster in the Tajik corpus is not visible, as it contains documents with perplexity values between 3,500 and 9,000.

	IS	ET	EU	GL	NE	TG
Documents (M)						
Unfiltered	2.07	6.94	1.56	4.55	2.94	1.28
Filtered	1.01	3.62	0.62	1.14	1.41	0.24
Remaining	48.7%	52.2%	39.9%	25.1%	48.0%	18.5%
Tokens (B)						
Unfiltered	1.12	3.04	0.58	1.21	0.97	0.56
Filtered	0.64	2.00	0.29	0.67	0.61	0.13
Remaining	57.3%	65.7%	50.1%	55.0%	63.5%	23.8%
Examples (M)						
Unfiltered	3.69	11.13	2.05	3.79	5.26	1.88
Filtered	1.75	6.51	0.86	1.72	1.56	0.36
Remaining	47.3%	58.5%	41.9%	45.4%	29.7%	18.9%

Table 4.7: The size of the mC4 corpus subsets before and after filtering with the GMM classifier, measured in millions of documents, billions of space-delimited tokens, and millions of pre-training examples. Percentages indicate the proportion of data retained post-filtering. Language codes: IS (Icelandic), ET (Estonian), EU (Basque), GL (Galician), NE (Nepali), TG (Tajik).

Table 4.7 provides an overview of the sizes of the mC4 subsets before and after filtering, measured in the number of documents, space-delimited tokens, and pre-training examples. The proportion of high-quality text retained after filtering varies significantly across languages, reflecting the varying degrees of noise in the unfiltered corpora. For example, the Estonian corpus retained the highest proportion of high-quality text, with 65.7% of space-delimited tokens remaining after filtering. In contrast, the Tajik corpus had the most significant quality issues, retaining only 23.8% of tokens.

4.5.3 Results

We pre-trained LMs on both the filtered and unfiltered corpora and evaluated their downstream performance on a benchmark of NLP tasks. As discussed in Section 3.1, the selected languages are diverse, both with respect to language family and available resources. The pre-training corpora range from approximately 130 million to over 3 billion space-delimited tokens in size, while the downstream datasets vary in complexity and scale, consisting of hundreds to millions of examples.

Filtering consistently resulted in similar or improved performance across all tasks in our evaluation. The downstream results are summarized in Table 4.8, which presents the scores for the filtered and unfiltered models, along with the absolute and relative changes in performance.

On average, filtering improved performance across all six languages by 0.66 percentage points, corresponding to a relative gain of 4.44%. However, the impact of filtering varied significantly by task and language:

Part-Of-Speech (POS) Tagging The improvement in tagging accuracy was minimal, with an average increase of 0.02 percentage points, or a relative gain of 0.93%. Statistically significant improvements were observed only for Icelandic, where accuracy increased by 0.11 percentage points. The unfiltered models already achieved a high

Language	Task	Unfiltered	Filtered	Δ Score	Rel. Change
Icelandic	POS	97.00%	**97.11%**	0.11%	3.67%
Icelandic	NER	91.38%	91.55%	0.17%	1.97%
Icelandic	DP	**84.95%**	84.85%	-0.10%	-0.66%
Icelandic	ATS	71.48%	72.12%	0.28%	0.99%
Icelandic	QA	59.23%	59.61%	0.38%	0.93%
Icelandic	Average	80.74%	80.98%	0.17%	0.88%
Estonian	POS	97.98%	98.00%	0.02%	0.99%
Estonian	NER	75.23%	75.60%	0.37%	1.49%
Estonian	DP	88.23%	**89.03%**	0.80%	6.80%
Estonian	Average	87.15%	87.54%	0.40%	3.09%
Basque	POS	97.05%	97.04%	-0.01%	-0.34%
Basque	NER	80.65%	**82.47%**	1.82%	9.41%
Basque	DP	84.38%	**84.86%**	0.48%	3.07%
Basque	Average	87.36%	88.12%	0.76%	6.04%
Galician	POS	98.93%	98.91%	-0.02%	-1.87%
Galician	NER	86.92%	87.40%	0.48%	3.67%
Galician	DP	85.95%	**86.59%**	0.64%	4.56%
Galician	Average	90.60%	90.97%	0.37%	3.90%
Nepali	POS	96.17%	96.19%	0.02%	0.52%
Nepali	NER	89.70%	**90.67%**	0.97%	9.42%
Nepali	TC	75.78%	78.82%	3.04%	12.55%
Nepali	Average	87.22%	88.56%	1.34%	10.51%
Tajik	NER	78.20%	81.14%	0.94%	4.52%
Tajik	TC	78.11%	**80.20%**	2.09%	9.55%
Tajik	Average	78.66%	80.17%	1.52%	7.10%
All	Average	85.19%	85.85%	0.66%	4.44%

Table 4.8: Downstream performance of TEAMS-Small models pre-trained on filtered and unfiltered corpora. Scores in **bold** indicate statistically significant differences between the filtered and unfiltered models (paired t-test; $p < 0.05$).

average tagging accuracy of 97.43%, suggesting limited opportunities for improvement without increasing the model size.

Dependency Parsing (DP) Filtering provided a modest increase in LAS scores, with an average improvement of 0.45 percentage points, corresponding to a relative gain of 3.22%. Statistically significant gains were observed for Estonian, Basque, and Galician, while the Icelandic score decreased by 0.10 percentage points. This may be due to the fact that the IcePaHC dataset contains a significant amount of historical Icelandic text. Such text may have been discarded during filtering, exposing the model to less historical text and negatively impacting performance on historical datasets.

Named Entity Recognition (NER) The benefit was more significant for NER, with an average improvement of 0.79 percentage points, for a relative gain of 4.90%. We observed a statistically significant improvement for Basque and Nepali.

Topic Classification (TC) The most significant improvement was in topic classification, where the average accuracy improved by 2.57 percentage points, for a relative gain of 11.13%. Despite the small size of the SIB-200 topic classification dataset, which consists of 1,000 examples for each language, the increase in accuracy is quite significant. One possible interpretation of these results is that larger downstream datasets might overcome weaknesses in pre-trained models through sheer volume of task-specific examples, thus masking the benefits of pre-training on higher-quality data.

We observed the greatest improvements from text quality filtering on datasets where the unfiltered models had low baseline scores. Filtering did not result in statistically significant gains for tasks where the unfiltered models achieved scores of 90% or higher. In such cases, the TEAMS-Small models may be more constrained by their size than by the quality of the pre-training data. For datasets with extremely high baseline scores (e.g., 98.93% for POS tagging in Galician), further improvement may be implausible due to annotation errors or inherent ambiguities in the data.

Overall, our results suggest that small encoder-only LMs are surprisingly robust to noise in pre-training corpora when evaluated on supervised classification tasks across the languages included in our study. Nevertheless, for more challenging tasks and datasets, text quality filtering is likely to yield a meaningful and positive impact on downstream results. However, care must be taken to ensure that the high-quality corpus used for feature extraction (e.g., training an n-gram model to calculate perplexity) is representative of the downstream datasets, or else filtering may inadvertently degrade performance.

The GMM-based text quality classifier proved effective for filtering corpora in all six languages. It significantly reduced each corpus in size while generally improving downstream performance. The classifier was trained using only three features, which were relatively simple and computationally inexpensive to extract. Once trained and after features had been extracted from the corpus, the classifier could label over 4.2 million documents per second, without relying on distributed computing or GPU acceleration.

Given these benefits and the relatively low cost of implementation, we conclude that text quality filtering is a valuable step when pre-training on web-crawled corpora. Even in cases where filtering has minimal or no positive impact on downstream performance, it can still lead to a much more computationally efficient pre-training process. As shown in Table 4.7, filtering reduced the number of pre-training examples by 41.5% to 81.1%, which can proportionally accelerate pre-training when the number of epochs is fixed. Alternatively, for a fixed compute budget, filtering results in more useful examples being included in every batch. To fully realize these benefits, it is crucial to ensure that the high-quality corpus is sufficiently representative in order to minimize the risk of discarding potentially useful text.

4.6 Conclusions

We have presented a new text quality dataset for Icelandic, TQ-IS, which consists of 2,000 documents that have been manually annotated, both with regard to overall document quality as well as by identifying and labeling low-quality text spans within each document. To the best of our knowledge, this is the first document-level dataset of its kind.

We have evaluated the effectiveness of a large number of heuristic rules commonly applied for text quality filtering, both individually and as part of a larger ruleset. We have demonstrated that perplexity is the most effective feature, by far, when distinguishing between low and high-quality documents. We have also shown that optimal results can be obtained with the use of only a handful of rules. Optimal rulesets and thresholds may differ between corpora and languages depending on their characteristics. However, we have shown that visualizing the distribution of documents by feature pairs can reveal close to optimal threshold values in an intuitive manner, avoiding time-consuming analysis, manual labeling, or guesswork.

We have evaluated four categories of text quality classifiers on the TQ-IS dataset, with the supervised classifier proving to be the most effective. We have shown that a very small LM with only 14M parameters can be fine-tuned to detect a wide range of low-quality text categories with close to perfect accuracy when trained on a small, manually labeled dataset. Our results also agree with previous findings that show perplexity to be a highly useful proxy for document quality, as long as the LM has been trained on a high-quality, representative corpus.

Furthermore, we have proposed a simple, but novel, approach to text quality filtering based on unsupervised clustering and outlier detection algorithms. In particular, we find that the results obtained with a GMM classifier are comparable to those achieved with a rule-based approach, even using an optimal ruleset and threshold derived from a manually labeled dataset. The key benefits of this approach is that it does not require time-consuming feature engineering or parameter optimization, the use of manually labeled data, or language expertise for the languages to be filtered.

We further evaluated the GMM classifier by using it to filter low-quality documents from web-crawled corpora and assessed its impact on downstream performance. Our experiments demonstrated that text quality filtering generally results in similar or improved performance on downstream tasks, particularly for more challenging datasets and tasks. This suggests that text quality classifiers based on clustering and outlier detection algorithms are effective across a wide range of low and medium-resource languages.

In addition to its benefits on downstream performance, text quality filtering increases the data efficiency of the pre-training process by reducing compute spent on low-quality examples. For a fixed number of epochs, filtering results in faster pre-training, while for a fixed compute budget, it allows the model to make more passes through high-quality data, improving learning efficiency. Even in cases where filtering has minimal impact on downstream performance, the increased computational efficiency makes it a valuable step in processing noisy pre-training corpora. We conclude that text quality filtering is an important component of the pre-training process, providing benefits for both model performance and computational efficiency.

Chapter 5

Multilingual Models

Leveraging multilingual data and models for low-resource languages is an established strategy in natural language processing (NLP). One of the most direct approaches involves pre-training language models (LMs) on multilingual corpora, as demonstrated by models such as mBERT (Devlin et al., 2019), XLM-R (Conneau et al., 2020), mT5 (Xue et al., 2021), and PaLM 2 (Anil et al., 2023). These models have been pre-trained on large-scale datasets covering over a hundred languages. However, training massively multilingual models comes with inherent challenges and trade-offs. One key limitation is that model capacity (i.e., the number of parameters in the model) is shared across all languages in a multilingual LM, whereas monolingual models dedicate their full capacity to a single language. Additionally, the pre-training corpora are highly imbalanced, with low-resource languages significantly underrepresented. For instance, in the XLM-R pre-training corpus, which covers 100 languages, the 42 least represented languages collectively account for less than 1% of the total token count. Similarly, in the mT5 pre-training corpus, which spans 107 language-script pairs (e.g., where Hindi is counted separately depending on whether it is written in Devanagari or Latin script), the 46 least represented pairs contribute only about 1% of the total tokens. This severe data imbalance negatively impacts downstream performance for low-resource languages (Conneau et al., 2020).

These issues can be mitigated using several strategies, such as increasing model capacity and employing data sampling techniques that prioritize low-resource languages during pre-training (Conneau et al., 2020; Xue et al., 2021). However, even with these adjustments, massively multilingual models often underperform compared to monolingual or bilingual models (Pyysalo et al., 2021; Snæbjarnarson et al., 2023). It has been suggested that to optimize downstream performance in a specific low-resource language, a more effective approach may be to pre-train with a small number of closely related, higher-resource languages rather than relying on broad multilingual pre-training (Wu and Dredze, 2020; Pyysalo et al., 2021; Snæbjarnarson et al., 2023).

Beyond multilingual pre-training, other techniques have been explored for improving cross-lingual transfer. One promising approach is *adapter-based methods*, as proposed by Houlsby et al. (2019). One prominent framework that employs this technique is MAD-X (Pfeiffer et al., 2020), which integrates three types of small, trainable *adapter modules* into the layers of a pre-trained model. *Language adapters* learn how to process specific languages, while *task adapters* learn language-agnostic knowledge for downstream tasks. A third type, *invertible adapters*, helps the model efficiently handle tokens from languages not seen during pre-training. A task adapter can first be trained on a task in a high-resource language, then reused with language adapters

for low-resource languages, leveraging what it learned about the task without requiring new task-specific training data. During training, only the adapter parameters are updated while the core model remains frozen, significantly reducing computational costs. The authors of MAD-X found that integrating their approach into XLM-R and mBERT led to substantial improvements in downstream tasks such as named entity recognition (NER) and causal commonsense reasoning across 16 diverse languages.

Another technique for cross-lingual transfer is *lexical adaptation*, as explored by Artetxe et al. (2020). This method first pre-trains a monolingual model on a source language and then transfers it to a new language by freezing all model parameters except for the embedding layer. The model is further trained on text from the target language using the same pre-training objective, allowing it to learn new lexical representations while preserving the original high-level language representations. This approach enables zero-shot cross-lingual transfer without requiring a shared vocabulary or joint multilingual training. However, lexical adaptation slightly underperforms compared to models pre-trained on bilingual corpora, with an average performance drop of 3.3 points on cross-lingual classification and question answering (QA) datasets. The difference is reduced to 1.1 points when employing additional techniques, such as language-specific embeddings and noised fine-tuning.

Beyond these approaches, several alternative strategies exist for cross-lingual transfer, including cross-lingual continual learning (M'hamdi et al., 2023), meta-learning (Gu et al., 2018), and multilingual knowledge distillation (Reimers and Gurevych, 2020). While these techniques offer promising results, they typically require specialized training pipelines, additional computational resources, or more complex model architectures.

In this chapter, we investigate the impact of augmenting a monolingual pre-training corpus with text from different sources. Specifically, we experiment with augmenting a subset of the Icelandic Gigaword Corpus (IGC) (Steingrímsson et al., 2018) with three distinct types of additional data: (1) Norwegian text, representing a closely related language; (2) Finnish text, representing an unrelated language; and (3) Python code, representing structured, non-linguistic data. The inclusion of Python code in pre-training corpora has been found to improve downstream performance on certain NLP tasks (Muennighoff et al., 2023). We pre-train an LM on each augmented corpus and evaluate its performance on a benchmark of Icelandic NLP tasks. This experiment aims to answer our second research question: **RQ2: How does linguistic similarity influence the effectiveness of cross-lingual transfer in bilingual models?** By including both an unrelated language and programming code as controls, we seek to distinguish whether performance gains stem from linguistic similarity or simply from exposure to a larger dataset.

Based on the results of this experiment, we apply the best-performing augmentation strategy to monolingual corpora for five low- to medium-resource languages. We then compare the downstream performance of bilingual models pre-trained on these augmented corpora with that of monolingual models. This evaluation allows us to assess whether the observed benefits generalize across a diverse set of languages.

While other multilingual learning approaches, such as adapter-based methods or lexical adaptation, may offer theoretical advantages, they often require architectural modifications or specialized training procedures. In contrast, augmenting pre-training corpora with additional text provides a straightforward and interpretable way to evaluate the impact of cross-lingual information on downstream performance. By systematically varying the type of additional data included in pre-training, we aim to better

understand the role of linguistic similarity and data diversity in improving language model performance for low-resource languages.

5.1 Related Work

XLM (Conneau and Lample, 2019) introduced the Translation Language Modeling (TLM) objective, which leverages parallel corpora to pre-train multilingual models. In this task, the model learns to predict masked tokens in parallel sentences by using contextual information from both languages. XLM, with 250M parameters, was pre-trained on Wikipedia articles in 15 languages, along with several parallel corpora. By combining Masked Language Modeling (MLM) on monolingual texts with TLM on parallel texts, classification accuracy on the XNLI natural language inference (NLI) benchmark (Conneau et al., 2018) was improved by 3.6 percentage points. To mitigate data imbalance, XLM employed a sampling strategy that increased the likelihood of selecting text from underrepresented languages during pre-training.

Building on this work, XLM-R (Conneau et al., 2020) adopted the RoBERTa pre-training approach (Liu et al., 2019), training for longer on larger datasets with increased batch sizes. The full model (550M parameters) was pre-trained on 295B tokens of web-crawled text across 100 languages, alongside a smaller XLM-R-Base variant (270M parameters). The sampling strategy was adjusted to more strongly favor low-resource languages, significantly improving their downstream performance while having minimal impact on high-resource languages. Observing that additional pre-training led to substantial performance gains, the authors concluded that XLM had been undertrained on the MLM task. Based on this finding, XLM-R exclusively used the MLM objective, omitting TLM. XLM-R achieved an average accuracy of 80.1 on XNLI, outperforming XLM by 5 percentage points.

However, while massively multilingual models generally achieve strong performance, there is ample research showing that they underperform on low-resource languages. Wu and Dredze (2020) evaluated mBERT on monolingual downstream tasks, including NER, part-of-speech (POS) tagging, and dependency parsing (DP), across 99 languages. While mBERT performed well for high-resource languages, its performance dropped significantly for low-resource languages. Notably, for the 30% least-represented languages, it was outperformed by BiLSTM-based methods, particularly in NER.

To further investigate the limitations of multilingual models, the authors pre-trained several monolingual and bilingual BERT models on Wikipedia articles for low-resource languages. Their results showed that the bilingual models generally outperformed monolingual models. Specifically, a Latvian-Lithuanian bilingual model achieved similar or better performance on Latvian tasks compared to a monolingual Latvian model, with a similar trend observed for an Afrikaans-Dutch model on Afrikaans tasks. Moreover, Pyysalo et al. (2021) found that monolingual BERT models pre-trained on Wikipedia articles frequently outperformed mBERT on monolingual DP tasks. The authors pre-trained 42 models, each on a different language, and found that 28 (or two-thirds) achieved a higher labeled attachment score (LAS) than mBERT. On average, monolingual models obtained a LAS score of 86.6, compared to 86.1 for mBERT.

More recently, Snæbjarnarson et al. (2023) explored the benefits of leveraging closely related languages for pre-training with ScandiBERT, a RoBERTa-Base model

(125M parameters) pre-trained on 96.8 GB of Scandinavian text: Icelandic (24.2%), Danish (7.1%), Norwegian (63.5%), Swedish (5.1%), and Faroese (0.1%). The model consistently outperformed XLM-R-Base on Faroese tasks, except for semantic similarity, supporting the hypothesis that pre-training on a small set of closely related languages is more effective than broad multilingual pre-training when targeting a specific low-resource language. Furthermore, their experiments showed that a monolingual Icelandic model outperformed a monolingual Danish model on Faroese tasks, which suggests that linguistic similarity plays a significant role in cross-lingual transfer.

Beyond leveraging related languages for pre-training, researchers have also explored cross-lingual transfer using non-linguistic data. Muennighoff et al. (2023) examined the impact of mixing English text with code. They replaced half of the text in an 84B token subset of the C4 corpus (Raffel et al., 2020) with Python code from The Stack (Kocetkov et al., 2023), a large dataset of permissively licensed code in 30 programming languages. Then, they pre-trained a 4.2B parameter GPT-2 model on this code-augmented corpus and compared it to an identical model pre-trained on the original C4 subset, which served as a baseline. Both models were pre-trained for a single epoch.

When evaluated on 19 natural language tasks, the code-augmented model outperformed the baseline by an average of 1.7 percentage points, with the largest gains in structured data-to-text generation, reading comprehension, and NLI tasks. Performance varied across other tasks, showing minor improvements in some and slight declines in others. Despite both models being pre-trained on the same total number of tokens (84B), supplementing 42B tokens of English text with code proved more beneficial than doubling the dataset with additional monolingual text. These results demonstrate the potential of non-linguistic augmentation strategies for generative models in high-resource settings and raise the question of whether similar benefits might extend to bidirectional encoder-only models pre-trained on low- and medium-resource languages. However, while programming code itself is non-linguistic, it often includes natural language comments that describe its function or provide additional context for readers. As noted by Kocetkov et al. (2023), the vast majority of these comments are in English, suggesting that code augmentation may provide disproportionate benefits for English-language tasks compared to other languages.

5.2 Experimental Setup

In this section, we describe how we augmented a subset of the IGC with text from three sources and evaluated the impact of each on downstream performance. Additionally, we describe how we augmented monolingual corpora for Icelandic, Estonian, Basque, Galician, and Nepali with text from related languages, and compared the performance of monolingual and bilingual models on a benchmark of NLP tasks.

5.2.1 Comparing Data Sources for Pre-Training Augmentation

To investigate the impact of different data sources on downstream performance when augmenting monolingual corpora, we supplemented a subset of the IGC with three types of additional text: Norwegian, a closely related language; Finnish, an unrelated language; and Python code. The core hypothesis of this experiment is that linguistic similarity improves cross-lingual transfer, meaning that augmenting an Icelandic cor-

pus with Norwegian text should yield better results than augmenting it with Finnish. Including Python code provided a contrast to natural language augmentation.

Below, we describe each donor language and its source dataset in detail:

- **Norwegian**, a Scandinavian language closely related to Icelandic. The documents were randomly sampled from FineWeb2[1] (Penedo et al., 2024), a large, multilingual, web-crawled corpus derived from the Common Crawl dataset that has been extensively filtered and deduplicated (we did not apply additional filtering). If the hypothesis holds, the Icelandic-Norwegian model should achieve performance comparable to a monolingual Icelandic model pre-trained on the full IGC.

- **Finnish**, a Finno-Ugric language that is unrelated to Icelandic. However, since both languages use the Latin script, script differences are eliminated as a confounding factor, allowing us to attribute any observed performance differences more clearly to linguistic similarity. Finnish text was also sampled from the FineWeb2 corpus. If the hypothesis is correct, the Icelandic-Norwegian model should outperform the Icelandic-Finnish model.

- **Python code** sampled from the deduplicated version of The Stack[2] (Kocetkov et al., 2023). When sampling code files, we ignored documents containing Chinese characters (e.g., in comments), as they otherwise took up a substantial portion of the subword tokenizer's vocabulary. Structured data of this kind has been shown to improve downstream performance on certain natural language tasks, such as reading comprehension and reasoning (Muennighoff et al., 2023). However, since our benchmark primarily consists of lower-level linguistic tasks (e.g., NER, POS, DP), it may not fully capture the potential benefits of code augmentation.

We randomly sampled documents from the IGC until we obtained a subset containing 837M tokens, half the size of the full corpus. We refer to this subset as *IGC-50*. We then created three bilingual versions of the corpus (Icelandic-Norwegian, Icelandic-Finnish, and Icelandic-Python), ensuring that the two languages were equally represented in each augmented corpus. Since word lengths vary significantly across the four languages, we balanced the corpora based on the number of non-whitespace characters rather than token count. For each language, we randomly sampled additional documents until the total number of non-whitespace characters in IGC-50 had been doubled. Table 5.1 provides details on corpus sizes.

By augmenting a trimmed version of the IGC, we were able to compare the three data sources directly while also evaluating their performance against monolingual models pre-trained on both versions of the IGC. This provides insights into the relative benefits of each augmentation source as well as the impact of adding more Icelandic text.

By doubling the size of IGC-50, we minimized language imbalance, eliminating the need for a sampling strategy during pre-training. Muennighoff et al. (2023) found that equal proportions of natural language and programming code yielded optimal results over other mixing ratios, supporting our approach. While different ratios might be more suitable when combining two natural languages, maintaining an even split allowed

[1]FineWeb2: `https://huggingface.co/datasets/HuggingFaceFW/fineweb-2`

[2]The Stack (deduplicated): `https://huggingface.co/datasets/bigcode/the-stack-dedup`

Corpus	Documents (M)	Tokens (B)	Examples (M)
IGC	5.13	1.67	4.38
IGC-50	2.56	0.84	2.19
IGC-50 + Norwegian	4.10	1.68	4.41
IGC-50 + Finnish	3.99	1.42	4.12
IGC-50 + Python	3.75	1.34	5.50

Table 5.1: Corpus sizes in their original and augmented forms. The IGC is shown in its full and trimmed versions (labeled IGC and IGC-50, respectively), with the latter augmented using Norwegian, Finnish, or Python code. Sizes are measured in millions of documents, billions of space-delimited tokens, and millions of pre-training examples.

for clearer interpretation of our hypothesis. With both languages equally represented, performance differences are more likely attributable to linguistic similarity rather than language imbalance. We leave the exploration of alternative mixing ratios for future work.

We trained a WordPiece tokenizer from scratch on each corpus, following the same process as for monolingual corpora. Since the number of unique word forms was significantly higher in the bilingual corpora than in the IGC, we increased the tokenizer's vocabulary size from 32k to compensate. We explored vocabulary sizes of 48k and 64k, evaluating the average number of subwords per word, the average number of characters per subword, and the proportion of words that the tokenizer did not need to split when tokenizing IGC-50, as shown in Table 5.2. A 48k vocabulary was sufficient for the Icelandic-Norwegian and Icelandic-Python corpora, while the Icelandic-Finnish corpus required 64k to match the performance of the Icelandic tokenizer. To ensure consistency and avoid introducing vocabulary size as a confounding factor, we used a 64k vocabulary for all bilingual corpora, while retaining 32k for monolingual models.

Metric	IS	IS-NO		IS-FI		IS-PY	
	32k	48k	64k	48k	64k	48k	64k
Avg. subwords per word	1.16	1.18	1.15	1.20	1.16	1.17	1.14
Mean subword length	3.89	3.83	3.94	3.78	3.89	3.85	3.97
Complete words (%)	88.8	87.5	89.5	86.6	88.6	88.0	90.1

Table 5.2: WordPiece tokenizer statistics when processing the IGC-50 corpus. Tokenizers were trained on IGC-50 alone (IS) or when augmented with Norwegian (IS-NO), Finnish (IS-FI), or Python code (IS-PY). Vocabulary sizes vary from 32k to 64k tokens. The complete words statistic represents the proportion of tokens that were not split into subwords by the tokenizer.

5.2.2 Augmenting Monolingual Pre-Training Corpora

To investigate whether linguistic similarity enables better cross-lingual transfer, we compared monolingual models against bilingual models pre-trained on related language pairs. For each of five diverse low- to medium-resource languages, we augmented monolingual data with text from FineWeb2. The additional text was sampled from

languages that were typologically related, mutually intelligible, or exerted historical influence on the target language.

For our experiments, we pre-trained TEAMS-Small models on the following bilingual corpora and compared their performance against monolingual baselines:

- **Icelandic** and **Norwegian**: North Germanic languages sharing significant lexical and grammatical similarities. We used Norwegian Bokmål, the most widely used written form of Norwegian, to augment the IGC.
- **Estonian** and **Finnish**: Finno-Ugric languages with substantial grammatical and lexical similarities, exhibiting partial mutual intelligibility (Gooskens and Härmävaara, 2019).
- **Basque** and **Spanish**: A typologically distinct pair, with Basque being a language isolate with no known relatives, and Spanish a Romance language. Despite their fundamental differences in structure (subject-object-verb vs. subject-verb-object), Spanish has exerted lexical influence on Basque through centuries of contact (Cenoz, 2000).
- **Galician** and **Portuguese**: Western Ibero-Romance languages descended from medieval Galician-Portuguese, sharing significant mutual intelligibility (Monteagudo, 2024). While Portuguese evolved independently and Galician was significantly influenced by Spanish, both maintain very similar grammar and vocabulary.
- **Nepali** and **Hindi**: Indo-Aryan languages descended from Sanskrit, sharing the Devanagari script, subject-object-verb word order, and many other grammatical characteristics. They exhibit significant lexical overlap, although Nepali uses additional characters not found in Hindi.

We doubled the size of each monolingual corpus by randomly sampling text from the corresponding donor language in FineWeb2. Table 5.3 shows the resulting sizes of both monolingual and bilingual corpora. For monolingual data, we used the IGC for Icelandic, the ENC (Koppel and Kallas, 2022) for Estonian, and EusCrawl (Artetxe et al., 2022) for Basque. For Galician and Nepali, for which curated pre-training corpora were not available, we used the filtered versions of their subsets in the mC4 corpus (Xue et al., 2021), as described in Section 4.5.2. All models used WordPiece tokenizers, with vocabulary sizes of 32k for monolingual models and 64k for bilingual models, maintaining consistency with our earlier experiments (see Section 5.2.1).

5.3 Results

In this section, we present the results of our experiments with augmenting monolingual corpora with text from a donor language.

5.3.1 Comparing Data Sources for Pre-Training Augmentation

We pre-trained TEAMS-Small models on several monolingual and bilingual versions of the IGC, with results summarized in Table 5.4. The monolingual model pre-trained

Corpus	Documents (M)	Tokens (B)	Examples (M)
Icelandic (IGC)	5.13	1.67	4.38
Icelandic + Norwegian	7.96	3.36	8.84
Estonian (ENC)	2.14	0.43	1.36
Estonian + Finnish	3.07	0.84	2.60
Basque (EusCrawl)	1.59	0.29	0.85
Basque + Spanish	2.30	0.67	1.79
Galician (mC4-GL)	1.14	0.67	1.72
Galician + Portuguese	2.52	1.34	3.37
Nepali (mC4-NE)	1.41	0.61	1.56
Nepali + Hindi	3.60	1.47	3.47

Table 5.3: Sizes of monolingual and bilingual corpora, measured in millions of documents, billions of space-delimited tokens, and millions of pre-training examples.

on the full IGC achieved the highest overall performance, consistently matching or outperforming all other models across tasks. Notably, the IGC-50 model, which was trained on half as much data but for twice the number of epochs (i.e., the same number of steps, but 58.4 epochs as opposed to 29.2 epochs), performed almost as well. In three out of five tasks, the performance differences between these two models were not statistically significant, suggesting that increasing the number of training steps can compensate for a smaller pre-training corpus up to a point, at least for a model of this size. These results align with findings by Muennighoff et al. (2023), who observed that, for a large generative model, repeated data was nearly as effective as new data for up to 4 epochs. Training beyond this point continued to yield improvements, though rapidly diminishing returns set in around 16 epochs. It is possible that this limit differs for encoder-only models, where examples can vary between epochs (e.g., due to dynamic token masking in the MLM task), potentially allowing models to benefit from additional repetition.

Languages	POS	NER	DP	ATS	QA	Avg
IS (100%)	**97.01%**	**91.48%**	**84.83%**	**72.65**	**60.03%**	81.20%
IS (50%) + NO	96.89%	**91.67%**	84.51%	72.45	59.66%	81.04%
IS (50%) + FI	96.86%	**91.44%**	84.55%	**72.69**	59.35%	80.98%
IS (50%) + PY	96.81%	91.02%	84.32%	72.29	59.50%	80.79%
IS (50%)	**97.02%**	91.20%	**84.83%**	**72.83**	59.20%	81.02%

Table 5.4: Downstream performance of TEAMS-Small models pre-trained on monolingual and bilingual versions of the IGC. Scores in **bold** are statistically indistinguishable from the best result for each task (paired t-test with Holm-Bonferroni correction; $p < 0.05$).

Overall, performance differences across models were minor, with the lowest and highest average scores differing by just 0.41 percentage points. This narrow range suggests that TEAMS-Small may not have sufficient capacity to fully leverage additional data or cross-lingual augmentation. However, the results indicate that bilingual augmentation does not significantly degrade performance. The Icelandic-Norwegian model, for example, achieved results comparable to the full monolingual model despite

being pre-trained on an equal mix of Icelandic and Norwegian text. This suggests that supplementing a monolingual corpus with text from a closely related language is a viable approach, though the advantages may be task-dependent. However, the findings do not provide strong evidence that linguistic similarity alone drives performance improvements. While the Norwegian-augmented model outperformed the Finnish-augmented model in overall average score, the difference was marginal at just 0.06 percentage points.

Augmenting the pre-training corpus with programming code did not yield any measurable benefits. The Python-augmented model had the lowest average score among the bilingual models, with no statistically significant improvements in any task. This aligns with findings from Muennighoff et al. (2023), which suggest that structured data primarily benefits tasks requiring reasoning and inference capabilities. Since our benchmark did not include such tasks, it may not have fully captured the potential advantages of code augmentation.

To further investigate the impact of pre-training corpus size and model capacity, we conducted a follow-up experiment under a more resource-constrained setting. This time, we sampled 100M tokens from the IGC and supplemented it with an equal amount of Norwegian, Finnish, or Python code. As a monolingual baseline, we also sampled a 200M token subset of the IGC. Additionally, we significantly reduced the training sets for POS tagging, NER, and DP to 5,000 sentences each, approximately 10–15% of their original size, while keeping validation and test sets unchanged. This setup better simulates a low-resource scenario, as the original datasets were relatively large.

To address the question of model capacity, we pre-trained TEAMS-Base models, which have significantly more parameters (110M) than TEAMS-Small (14M). Given the smaller pre-training corpora, we reduced the number of pre-training steps from the default 1M to 100k. This resulted in approximately 50 epochs of pre-training for all models, except the Python-augmented one, which was trained for 40 epochs. Since Python code contains a high number of symbols and operators that are tokenized separately, it produces significantly more subwords than natural language text, leading to a greater number of pre-training examples. For fine-tuning, we used the same hyperparameters as for the TEAMS-Small models, with the exception of reducing the number of epochs for POS tagging to 10, for QA to 3, and for ATS to 1. The results are presented in Table 5.5.

Languages	POS	NER	DP	ATS	QA	Avg
IS (200M)	**95.11%**	85.92%	**80.85%**	**72.93**	**61.53%**	79.27%
IS (100M) + NO	94.59%	**86.19%**	80.53%	72.57	**61.36%**	79.05%
IS (100M) + FI	94.71%	84.61%	80.52%	72.26	60.79%	78.58%
IS (100M) + PY	94.37%	84.46%	80.11%	72.59	60.28%	78.36%

Table 5.5: Downstream performance of TEAMS-Base models pre-trained on augmented corpora. Scores in **bold** are statistically indistinguishable from the best result for each task (paired t-test with Holm-Bonferroni correction; $p < 0.05$).

The performance differences between the TEAMS-Base models were much clearer than those observed in the TEAMS-Small experiment. In four out of five tasks, one model significantly outperformed all others. As in the previous experiment, the monolingual model achieved the highest overall performance, followed closely by the

Norwegian-augmented model, which obtained the best score on NER and matched the top-performing model in QA. Among the bilingual models, the Norwegian-augmented model outperformed the Finnish-augmented one, achieving an average score that was 0.47 percentage points higher, a much larger difference than in the TEAMS-Small results. These findings support the hypothesis that linguistic similarity enhances cross-lingual transfer, as the model pre-trained with Norwegian text exhibited stronger performance than the one augmented with Finnish.

5.3.2 Augmenting Monolingual Pre-Training Corpora

We augmented monolingual corpora for Icelandic, Estonian, Basque, Galician, and Nepali with text from a donor language. Whereas the previous experiments involved supplementing a subset of the IGC with text from other languages, here we used them to double the size of full monolingual corpora, as shown in Table 5.3. We then compared the downstream performance of TEAMS-Small models pre-trained on the monolingual and bilingual corpora. The results are summarized in Table 5.6.

Language	Task	Monolingual	Bilingual	Δ Score	Rel. Change
Icelandic	POS	**97.01%**	96.88%	-0.13%	-4.35%
Icelandic	NER	91.48%	91.68%	0.20%	2.35%
Icelandic	DP	**84.83%**	84.62%	-0.21%	-1.38%
Icelandic	ATS	72.65%	72.80%	0.15%	0.55%
Icelandic	QA	60.03%	59.83%	-0.20%	-0.50%
Icelandic	Average	81.20%	81.16%	-0.04%	-0.20%
Estonian	POS	**98.04%**	97.99%	-0.05%	-2.55%
Estonian	NER	78.13%	77.52%	-0.61%	-2.79%
Estonian	DP	**89.40%**	89.14%	-0.26%	-2.45%
Estonian	Average	88.52%	88.22%	-0.31%	-2.67%
Basque	POS	**97.00%**	96.86%	-0.14%	-4.67%
Basque	NER	84.42%	83.81%	-0.61%	-3.92%
Basque	DP	84.45%	84.38%	-0.07%	-0.45%
Basque	Average	88.62%	88.35%	-0.27%	-2.40%
Galician	POS	98.92%	**98.93%**	0.01%	0.93%
Galician	NER	87.40%	87.66%	0.26%	2.06%
Galician	DP	**86.59%**	86.02%	-0.57%	-4.25%
Galician	Average	90.97%	90.87%	-0.10%	-1.11%
Nepali	POS	96.19%	96.13%	-0.06%	-1.57%
Nepali	NER	90.67%	90.82%	0.15%	1.61%
Nepali	TC	78.82%	79.07%	0.25%	1.18%
Nepali	Average	88.56%	88.67%	0.11%	0.99%
All	Average	86.83%	86.71%	-0.11%	-1.19%

Table 5.6: Downstream performance of TEAMS-Small models pre-trained on monolingual and bilingual corpora for several languages. Scores in **bold** indicate statistically significant differences between the monolingual and bilingual models (paired t-test; $p < 0.05$).

The performance differences between monolingual and bilingual models were statistically significant in only 7 out of 17 tasks, and even in those cases, the differences

were small. On average, bilingual augmentation resulted in a negligible decrease of 0.11 percentage points in average score, suggesting that supplementing monolingual pre-training corpora with an equal amount of text from a donor language has minimal impact on downstream performance for small models, indicating that bilinguality may come at a low performance cost. This holds true even when the donor language is closely related to the target language.

While these findings indicate that monolingual training is generally preferable for the tasks evaluated, they do not rule out the potential benefits of multilingual augmentation in other contexts. One likely exception is for tasks that inherently require cross-lingual knowledge, such as machine translation or language classification, which were not included in this benchmark. Another scenario in which multilingual augmentation may be beneficial is when monolingual corpora are too small to effectively pre-train an LM. In such cases, leveraging text from a related language or continued pre-training with existing LMs can improve performance (Snæbjarnarson et al., 2023).

Although the overall effect of data augmentation was limited, the results suggest that linguistic similarity plays a role in cross-lingual transfer. Bilingual models trained on closely related languages (e.g., Galician and Portuguese) showed smaller performance drops compared to those trained on unrelated pairs (e.g., Basque and Spanish). However, the differences were modest, reinforcing the conclusion that the benefits of bilingual augmentation are highly task-specific.

For small models focused on monolingual tasks, these findings suggest that multilingual augmentation is generally unnecessary and may even slightly degrade downstream performance. However, it may still prove useful for tasks that involve multiple languages, settings where monolingual data is severely limited, or for larger models that might be able to better leverage additional training data.

5.4 Conclusions

In this chapter, we addressed our second research question: **RQ2: How does linguistic similarity influence the effectiveness of cross-lingual transfer in bilingual models?** Our findings indicate that linguistic similarity plays a minor role in cross-lingual transfer for small models, with monolingual pre-training generally yielding equal or slightly better performance on the evaluated tasks. Notably, the performance differences between monolingual and bilingual models were minimal, with bilingual models typically showing only a slight decrease in score. This suggests that for small models focused on low-level linguistic tasks (e.g., POS tagging, NER, and DP), monolingual pre-training is the more effective approach. A plausible explanation for these results is that smaller models lack the capacity to effectively leverage cross-lingual augmentation. However, it is still surprising that even though the models were not able to take full advantage of the additional data, neither did it appear to cause a significant disadvantage. That said,, our experiments did not include multilingual tasks (e.g., machine translation or language classification) or corpora that were insufficiently large for pre-training, both of which are settings where bilingual augmentation may provide stronger benefits.

The impact of linguistic similarity in bilingual models was inconsistent across our experiments. In some cases, such as with the Icelandic-Norwegian and Galician-Portuguese models, linguistic similarity appeared to contribute to a reduced performance cost compared to what we observed for models trained on less related language

pairs. However, this pattern did not hold universally, since the Estonian-Finnish model showed a larger performance drop than the Basque-Spanish model, despite Estonian and Finnish being closely related and partially mutually intelligible (Gooskens and Härmävaara, 2019), while Basque and Spanish are linguistically unrelated. This suggests that linguistic similarity alone is not a strong predictor of improved cross-lingual transfer in small bilingual models. Instead, the results indicate that for smaller models, the benefits of bilingual training, if any, remain marginal regardless of language similarity.

However, our experiments with larger models suggest that model capacity is a key factor in the effectiveness of bilingual training. With the larger TEAMS-Base models, language similarity had a much greater impact, as seen in the Icelandic-Norwegian model significantly outperforming the Icelandic-Finnish model. This aligns with results obtained by Wu and Dredze (2020), who found that for Latvian and Afrikaans tasks, monolingual BERT-Base models were consistently outperformed by Latvian-Lithuanian and Afrikaans-Dutch models with the same architecture. Similarly, Snæbjarnarson et al. (2023) observed meaningful benefits from language similarity when evaluating RoBERTa-Base models trained on Scandinavian languages on Faroese tasks. This suggests that larger models may be better equipped to leverage linguistic similarity for cross-lingual transfer, while smaller models struggle to take full advantage of multilingual augmentation due to limited capacity.

Chapter 6

Subword Tokenization

Subword tokenization addresses the challenge of handling words that a model has never seen before. Unlike word-level tokenizers, which treat each word as an indivisible unit, subword tokenization breaks unfamiliar words into smaller, known components. This approach is particularly beneficial for morphologically rich languages, where corpora often contain millions of unique word forms, many of which appear only once.

A key advantage of subword tokenization is its ability to maintain a fixed-size vocabulary while effectively representing an unlimited number of possible words. By decomposing rare or unknown words into familiar subwords, this method significantly mitigates the data sparsity issue inherent in traditional word-level tokenization. Models learn more efficiently because they encounter each subword more frequently, reducing, if not eliminating, out-of-vocabulary (OOV) words. These benefits have made subword tokenization a standard technique in pre-trained language models (LMs).

Subword tokenization algorithms aim to balance vocabulary size and coverage, ensuring that common words remain intact while splitting rare or unseen words into a minimal number of subwords. They achieve this by identifying frequent patterns in the training data and using them to encode text in a compact and efficient manner.

The vocabulary size, specified during training, directly influences how the tokenizer segments words. A smaller vocabulary reduces memory requirements, but results in more aggressive word splitting and longer token sequences. Conversely, a larger vocabulary preserves more complete words, but requires more memory. Table 6.1 illustrates this trade-off using an Icelandic news headline processed with WordPiece tokenizers (Wu et al., 2016) trained on the Icelandic Gigaword Corpus (IGC) (Steingrímsson et al., 2018) at different vocabulary sizes.

Tokenizer	#Tokens	Tokenized text
WordPiece 8k	8	`Se·x·tíu flug·ferð·um afl·ýst`
WordPiece 16k	6	`Sex·tíu flug·ferðum afl·ýst`
WordPiece 32k	5	`Sex·tíu flug·ferðum aflýst`
WordPiece 64k	4	`Sex·tíu flugferðum aflýst`
WordPiece 96k	3	`Sextíu flugferðum aflýst`

Table 6.1: Impact of vocabulary size on tokenizing the Icelandic headline *Sextíu flugferðum aflýst* ("Sixty flights canceled") using WordPiece. The dot (·) denotes subword boundaries within words.

Although subword tokenizers aim to minimize the number of splits, they may still encounter OOV words that must be decomposed into the smallest available subwords. In models such as BERT (Devlin et al., 2019), ALBERT (Lan et al., 2019), and DistilBERT (Sanh et al., 2020), the smallest vocabulary units are individual characters. However, since vocabulary size is limited, not all Unicode characters can be included. When encountering sequences of unseen characters that cannot be broken down into smaller subwords, some tokenizers might replace them with a special unknown token, significantly limiting the model's ability to capture semantic information. This is especially common for scripts that are absent or underrepresented in the training data. For example, Unicode defines over 100,000 Chinese characters, several thousand of which see regular use. Including all of them in a tokenizer's vocabulary would be extremely inefficient for models not primarily designed to process Chinese text. As a result, many such characters are replaced by the unknown token.

The need for the unknown token can be avoided altogether by representing the minimal vocabulary using individual bytes rather than characters. Any Unicode character can be encoded as a sequence of one to four UTF-8 bytes, each taking one of 256 possible values, forming the smallest possible vocabulary. This approach, popularized by RoBERTa (Liu et al., 2019), has since been widely adopted by models such as GPT-2 and its successors (Radford et al., 2019; Brown et al., 2020; OpenAI, 2024), DeBERTa (He et al., 2021), and ModernBERT (Warner et al., 2024). Alternative byte-level methods also exist, such as *byte-level fallback*, where the tokenizer primarily operates on the character level, falling back to a byte representation only for unseen sequences. While these techniques eliminate the need for an unknown token, they do not always lead to improved results and may even degrade downstream performance (Liu et al., 2019; Reimers and Gurevych, 2020).

In resource-constrained settings, tokenization choices have a more pronounced impact, as there is less data to compensate for suboptimal configurations. This makes tokenization particularly critical for low- to medium-resource languages. Additionally, it is often argued that subword tokenizers should ideally segment words along morphological boundaries, especially in morphologically rich languages. The reasoning is that encoding meaningful morphological units, rather than arbitrary substrings, could help models capture semantic information more effectively (Bostrom and Durrett, 2020). However, empirical findings on this hypothesis remain inconclusive. Some studies report improvements in specific downstream tasks (Hofmann et al., 2021), while others find that morphology-aware tokenization underperforms compared to traditional subword methods (Toraman et al., 2023; Kaya and Tantuğ, 2024).

In this chapter, we pre-train TEAMS-Small models on the IGC using WordPiece, BPE, and Unigram tokenizers with varying vocabulary sizes. We also compare the downstream performance of models trained on the IGC using character-level and byte-level BPE tokenizers. Finally, we take the tokenizer configuration that achieved the best overall performance on the IGC and use it to train tokenizers for Estonian, Basque, Galician, Nepali, and Tajik. We then pre-train TEAMS-Small models for these languages and compare their downstream performance against models trained with a WordPiece tokenizer using a 32k vocabulary. Our goal is to determine whether the trends observed for Icelandic hold for other low- and medium-resource languages. This investigation directly addresses our third research question: **RQ3: How do different subword tokenization algorithms and vocabulary sizes impact downstream performance for low- and medium-resource languages?**

While previous studies have examined the impact of subword tokenization algorithms and vocabulary sizes, most have focused on individual high-resource languages or were limited in scope. Many evaluated different tokenization strategies at a fixed vocabulary size or tested varying vocabulary sizes within a single algorithm. In contrast, our work systematically explores the effects of different tokenization strategies on downstream performance across six morphologically rich, low- and medium-resource languages. Our findings provide practical guidance for tokenizer selection in resource-constrained settings.

6.1 Subword Tokenization Algorithms

This section provides a brief overview of the subword tokenization algorithms used in our experiments, each of which takes a different approach to balancing vocabulary size and coverage.

6.1.1 Byte-Pair Encoding (BPE)

The BPE algorithm (Sennrich et al., 2016) is widely used in models such as RoBERTa, OpenAI's GPT series, and ModernBERT. It starts with a vocabulary consisting of the smallest elements in the training corpus, typically either individual characters or bytes. The algorithm iteratively expands the vocabulary by merging the most frequent adjacent subword pair at each step. The process continues until the vocabulary reaches a predefined size.

6.1.2 WordPiece

WordPiece (Wu et al., 2016) is used in models such as BERT, ELECTRA (Clark et al., 2020), and DistilBERT. While it also builds a vocabulary by iteratively merging subword units, it differs from BPE in its selection criterion. Instead of choosing the most frequent pair, WordPiece selects the pair that maximizes the increase in log-likelihood across the training corpus.

6.1.3 Unigram

The Unigram algorithm (Kudo, 2018) is used in models such as ALBERT, XLNet (Yang et al., 2019), and T5 (Raffel et al., 2020). Unlike BPE and WordPiece, which progressively build their vocabularies through merging, Unigram follows a subtractive approach. It starts with a large vocabulary containing words and frequent substrings and gradually reduces it to a target size. The tokenizer assigns probabilities to subword sequences using a unigram language model. During training, it evaluates the impact of each subword's removal on the overall corpus likelihood. A percentage of the least impactful subwords is iteratively pruned until the vocabulary reaches the desired size.

6.2 Related Work

6.2.1 Tokenization Algorithms

Several studies have compared subword tokenization methods in terms of their impact on downstream performance.

Bostrom and Durrett (2020) examined RoBERTa-Base models pre-trained on English and Japanese corpora using BPE and Unigram tokenizers, both with a vocabulary size of 20k. The Unigram tokenizer consistently outperformed BPE across four downstream tasks. In English question answering (QA), the Unigram model achieved an F_1 score 1.1 points higher than the BPE model, while in Japanese QA, the gap widened to 12.3 points in F_1. The authors attribute this difference to the Unigram tokenizer's ability to better handle languages without explicit word boundaries, such as Japanese, and its greater tendency to align subwords with linguistic morphology. Similarly, Zhang et al. (2020) found that for English abstractive summarization, Unigram yielded comparable or slightly better ROUGE scores than BPE across four datasets. Ali et al. (2024) compared BPE and Unigram using a 2.6B parameter decoder-only model. In an English monolingual setting, Unigram outperformed BPE on a benchmark of NLP tasks by 0.24 percentage points. For a multilingual model trained on German, French, Italian, Spanish, and English text, BPE achieved an average score 0.17 percentage points higher than Unigram. However, performance varied by language, with Unigram performing better for French and Spanish, and BPE for the other languages.

Toraman et al. (2023) investigated BPE and WordPiece tokenizers for Turkish RoBERTa models. Across six downstream tasks, WordPiece performed similarly to or slightly better than BPE, suggesting that while these methods differ algorithmically, their practical impact on downstream performance may be minimal in some cases.

Beyond these comparisons, several studies have evaluated byte-level tokenization. Qarah and Alsanoosy (2024) pre-trained BERT-Base models on an Arabic corpus using WordPiece, Unigram, and byte-level BPE tokenizers, each with a 50k vocabulary. They evaluated the models on seven tasks across 29 datasets, for a total of 36 different dataset-task combinations (with some datasets annotated for multiple tasks). The Unigram model achieved the best performance in 21 out of 36 evaluations, surpassing WordPiece by 0.77 percentage points on average. Despite the fact that byte-level BPE obtained the highest compression ratio, it underperformed against the other two algorithms on most tasks, obtaining an average score 0.95 percentage points lower than WordPiece. Liu et al. (2019) similarly found that RoBERTa models pre-trained with byte-level BPE performed slightly worse on English NLP tasks compared to character-level BPE. Zhang et al. (2022) investigated byte-fallback in a Unigram tokenizer and observed that excessive segmentation into byte units resulted in longer token sequences and, in many cases, degraded translation performance. These findings suggest that while byte-based tokenization ensures full character coverage, this benefit does not necessarily translate into improved downstream performance.

A separate line of work has explored morphological tokenization, which aims to segment words based on their internal structure. Toraman et al. (2023) pre-trained a Turkish RoBERTa model with a morphological tokenizer that split words into morphemes using a morphological analyzer. While this tokenizer performed competitively on some tasks, it generally underperformed compared to BPE and WordPiece. The authors noted that errors introduced by the analyzer may have contributed to these results. Given that accurate analyzers are often unavailable or difficult to develop in

low-resource settings, this raises concerns about the feasibility of morphological tokenization in such cases. Kaya and Tantuğ (2024) proposed an alternative morphological approach that extracts word roots and appends suffix tags as subwords. However, for a Turkish BERT-Base model, this method performed similarly or worse than a standard WordPiece tokenizer.

Overall, these studies indicate that Unigram tokenizers often outperform BPE or WordPiece, particularly in languages with complex morphology or ambiguous word boundaries. In contrast, byte-level tokenization tends to produce longer token sequences, which can negatively impact performance. Meanwhile, morphological tokenization remains an open research question, with mixed results suggesting that potential linguistic benefits of morpheme-based segmentation may be offset by practical challenges in implementation and the robustness of existing subword methods.

6.2.2 Vocabulary Size

Beyond the choice of tokenization algorithm, vocabulary size plays a crucial role in model performance by influencing the balance between vocabulary coverage and memory usage.

Zhang et al. (2020) explored the effect of vocabulary size in English abstractive summarization, comparing Unigram tokenizers with vocabulary sizes ranging from 32k to 256k. They found that ROUGE scores improved up to a vocabulary size of 96k, after which performance began to degrade, suggesting diminishing returns beyond a certain threshold.

For morphologically rich languages, larger vocabularies have shown a similar trend of initial performance gains followed by diminishing improvement. Toraman et al. (2023) evaluated vocabulary sizes for BPE, WordPiece, morphological, and word-level tokenizers in a Turkish RoBERTa model, finding that increasing vocabulary size generally improved downstream performance. However, for BPE and WordPiece, both of which outperformed the other tokenizers, the gains were minimal beyond 66k. The authors attributed this to increased vocabulary coverage and a reduction in OOV tokens. Ali et al. (2024) examined the impact of different vocabulary sizes (33k, 50k, 82k, and 100k) for BPE and Unigram tokenizers on downstream performance. For an English monolingual model, a vocabulary size of 33k yielded the highest average score, while larger vocabularies led to slightly lower performance. In contrast, for a multilingual model covering German, French, Italian, Spanish, and English, the best results were achieved with a vocabulary size of 100k, roughly three times larger than the optimal size for the monolingual model. However, the optimal vocabulary size differed between languages in the multilingual setting, with 33k performing best for German and 82k for English.

A more fine-grained analysis was conducted by Kaya and Tantuğ (2024), who varied vocabulary size in WordPiece tokenization for Turkish NLP tasks using a BERT-Base model. Increasing the vocabulary size from 32k to 64k led to substantial improvements, with named entity recognition (NER) scores improving by 1.6 percentage points and QA scores by up to 6.2 percentage points. However, increasing the vocabulary further to 128k and 256k provided only marginal additional gains. The authors suggested that reducing OOV words improves performance by limiting the overall number of token splits, which in turn reduces the likelihood of splits occurring at arbitrary positions rather than at morphological boundaries, potentially improving the model's ability to capture semantic information effectively.

Taken together, these studies suggest that increasing vocabulary size improves model performance up to a certain point, after which returns diminish. While larger vocabularies reduce OOV words, excessively large vocabularies may introduce inefficiencies without significant downstream benefits.

6.3 Experimental Setup

We trained WordPiece and BPE tokenizers using the *tokenizers* library for Python (Moi and Patry, 2023). For character-level tokenizers, we limited single-character subwords to 1,000 to prevent rare characters, such as emojis and mixed-script Unicode characters, from taking up too much of the vocabulary. We normalized the text by removing all Unicode control characters and split on whitespace characters and punctuation marks.

We trained Unigram tokenizers using the SentencePiece library (Kudo and Richardson, 2018) with settings that roughly matched the configuration we used for the WordPiece and BPE tokenizers. During training, we adjusted the character coverage of the tokenizer to restrict the number of single-character subwords to approximately 1,000, and replaced the default NFKC Unicode normalization with the less aggressive NFC normalization, converting all Unicode characters to their canonical forms. Due to the high memory usage when training with SentencePiece, we limited the training corpus for Icelandic to approximately 2M documents or 654M tokens, and for Nepali to 800k documents or 348M tokens (owing to the greater number of bytes required to represent characters in the Devanagari script).

For all tokenizers, we maintained original casing and retained all accents, and only used tokens that appeared at least twice for training.

6.4 Results

In this section, we present the results of our experiments with different tokenization strategies across several low- and medium-resource languages.

6.4.1 Tokenization Configurations for Icelandic

To assess the impact of different tokenization strategies, we pre-trained TEAMS-Small models on the IGC using WordPiece, BPE, and Unigram tokenizers, each with vocabulary sizes of 16k, 32k, and 64k. Table 6.2 provides statistics for each tokenizer, including the number of tokens generated from the corpus, the compression ratio (the ratio of non-whitespace characters to tokens), and the number of pre-training examples.

Larger vocabularies lead to fewer tokens and a higher compression ratio, which in turn affects the number of input sequences extracted from the corpus for pre-training. For example, a WordPiece tokenizer with a 64k vocabulary can represent the IGC with 4.13M examples (with each example consisting of 512 tokens), which is 13.6% fewer than required with a 16k vocabulary. With a fixed number of training steps, this effectively means that models using tokenizers with larger vocabularies are trained for more epochs. In 500k steps, a TEAMS-Small model will learn from each example 31 times with a WordPiece vocabulary of 64k, but only 26.8 times with a 16k vocabulary. Increasing the vocabulary size may therefore allow the model to learn more effectively during pre-training.

Tokenizer	Tokens	Compression ratio	Examples	Epochs
WordPiece 16k	2,431,605,858	3.56	4,777,862	26.8
WordPiece 32k	2,230,542,055	3.88	4,383,620	29.2
WordPiece 64k	2,101,252,609	4.12	4,130,112	31.0
BPE 16k	2,381,058,369	3.64	4,804,454	26.6
BPE 32k	2,179,331,643	3.98	4,400,810	29.1
BPE 64k	2,107,125,805	4.11	4,141,648	30.9
Unigram 16k	2,314,717,161	3.74	4,673,446	27.4
Unigram 32k	2,132,663,124	4.06	4,307,522	29.7
Unigram 64k	2,072,348,985	4.18	4,073,454	31.4

Table 6.2: Statistics for each tokenizer, showing the number of tokens generated from the IGC, the compression ratio, number of pre-training examples generated, and number of epochs at 500k pre-training steps.

Tokenizer	POS	NER	DP	ATS	QA	Avg
WordPiece 16k	96.99%	91.01%	84.62%	72.22	57.32%	80.43%
WordPiece 32k	97.01%	91.48%	**84.83%**	**72.65**	60.03%	81.20%
WordPiece 64k	97.06%	**91.97%**	**84.86%**	72.23	**60.68%**	81.36%
BPE 16k	96.54%	90.93%	84.50%	72.17	57.66%	80.36%
BPE 32k	96.90%	91.26%	84.73%	**72.87**	58.59%	80.87%
BPE 64k	97.01%	**91.64%**	84.73%	**72.76**	**60.32%**	81.29%
Unigram 16k	97.04%	90.80%	84.27%	72.47	58.62%	80.64%
Unigram 32k	**97.13%**	91.50%	84.76%	72.24	59.67%	81.06%
Unigram 64k	**97.17%**	**92.13%**	**84.84%**	**72.60**	**60.62%**	81.47%

Table 6.3: Downstream performance of TEAMS-Small models pre-trained on the IGC with different subword tokenizers and vocabulary sizes. Scores in **bold** are statistically indistinguishable from the best result for each task (paired t-test with Holm-Bonferroni correction; $p < 0.05$).

Table 6.3 presents the downstream performance of each model across five NLP tasks: part-of-speech (POS) tagging, NER, dependency parsing (DP), automatic text summarization (ATS), and QA. The results show that Unigram tokenization with a 64k vocabulary achieved the highest average score, slightly outperforming WordPiece and BPE at the same vocabulary size. This aligns with prior research suggesting that the Unigram algorithm has an advantage on some tasks (Bostrom and Durrett, 2020; Zhang et al., 2020; Qarah and Alsanoosy, 2024). Additionally, WordPiece performs similarly or slightly better than BPE, consistent with previous findings by Toraman et al. (2023).

Across all tokenization algorithms, larger vocabularies tend to yield better downstream performance, consistent with findings by Toraman et al. (2023) and Kaya and Tantuğ (2024). The effect is particularly pronounced for NER and QA, where increasing the vocabulary from 32k to 64k improved scores by an average of 0.50 and 1.11 percentage points, respectively. A greater compression ratio allows more characters to fit within each 512-token sequence, effectively increasing the amount of textual con-

text available to the model. Tasks that require a broader context window, such as QA, likely benefit from this effect, as they rely on longer-range dependencies.

Overall, these results suggest that while the Unigram algorithm offers a modest advantage, vocabulary size has a greater impact on downstream performance.

6.4.2 Byte-Level Tokenization

To evaluate the impact of byte-level tokenization on downstream performance, we pre-trained a TEAMS-Small model on the IGC using a byte-level BPE tokenizer with a 64k vocabulary, comparing it against a character-level tokenizer with the same vocabulary size. Table 6.4 provides statistics for each tokenizer.

Tokenizer	Tokens	Compression ratio	Examples	Epochs
Character-level BPE 64k	2,107,125,805	4.11	4,141,648	30.9
Byte-level BPE 64k	2,099,699,308	4.13	4,127,307	31.0

Table 6.4: Statistics for each tokenizer, showing the number of tokens generated from the IGC, the compression ratio, number of pre-training examples generated, and number of epochs at 500k pre-training steps.

There were minimal differences between the two tokenizers in terms of compression ratio and the number of examples generated. This is likely due to the fact that we used a tokenizer with a large vocabulary to process a curated, high-quality monolingual corpus. The character-level tokenizer generated 2.1 billion tokens when processing the IGC, of which only 5,475 were unknown tokens (i.e., sequences of characters that did not exist in its vocabulary). As such, there were negligible benefits from representing unknown characters as bytes. Since the tokenizer has a large vocabulary, a considerable portion of its capacity is allocated to byte-level merges that provide no practical advantage in this setting.

Tokenizer	POS	NER	DP	ATS	QA	Avg
Character-level BPE 64k	97.01%	91.64%	84.73%	72.76	60.32%	81.29%
Byte-level BPE 64k	96.99%	91.84%	84.32%	72.55	60.64%	81.27%

Table 6.5: Downstream performance of TEAMS-Small models pre-trained the IGC with character-level and byte-level BPE with a 64k vocabulary. Scores in **bold** indicate statistically significant differences between the two tokenizers (paired t-test; $p < 0.05$).

Table 6.5 presents the downstream performance of the character-level and byte-level tokenizers. The two tokenizers performed nearly identically across all tasks, with no statistically significant differences in downstream performance.

For high-quality monolingual corpora, byte-level tokenization does not appear to offer any advantages. Even in noisier corpora containing a variety of Unicode characters or mixed scripts, representing such noise as bytes rather than unknown tokens may not impact downstream performance, unless the downstream datasets contain similar noise distributions. The primary advantages of byte-level tokenization are more apparent in massively multilingual models, where vocabulary constraints make character-level representation impractical, and in generative models intended to be capable of outputting any Unicode character.

6.4.3 Cross-Linguistic Evaluation

For Icelandic, the Unigram tokenizer with a vocabulary size of 64k obtained the highest overall performance. To determine whether the same trends extend to other languages, we pre-trained TEAMS-Small models using a tokenizer with the same configuration on Estonian, Basque, Galician, Nepali, and Tajik corpora. Table 6.6 provides statistics for each tokenizer.

Language	Tokenizer	Tokens	Comp. ratio	Examples	Epochs
Icelandic	WordPiece 32k	2,230,542,055	3.88	4,383,620	29.2
Icelandic	Unigram 64k	2,072,348,985	4.18	4,073,454	31.4
Estonian	WordPiece 32k	689,653,932	4.06	1,356,437	94.4
Estonian	Unigram 64k	619,783,799	4.52	1,219,454	105.0
Basque	WordPiece 32k	430,028,142	4.55	846,250	151.3
Basque	Unigram 64k	401,787,770	4.86	790,913	161.8
Galician	WordPiece 32k	876,478,698	3.95	1,720,684	74.4
Galician	Unigram 64k	821,299,497	4.21	1,612,613	79.4
Nepali	WordPiece 32k	794,657,962	4.25	1,560,859	82.0
Nepali	Unigram 64k	743,490,989	4.54	1,460,580	87.6
Tajik	WordPiece 32k	181,492,212	4.27	356,308	359.2
Tajik	Unigram 64k	171,882,054	4.51	337,485	379.3

Table 6.6: Statistics for the two tokenizers for each language, showing the number of tokens generated from the pre-training corpus, the compression ratio, number of pre-training examples generated, and number of epochs at 500k pre-training steps.

We observed an improvement in compression ratio across all languages, with the largest gain in Estonian, where the pre-training corpus could be encoded with 10% fewer tokens. For Icelandic, the Unigram tokenizer achieved a compression ratio of 4.18, substantially higher than the 3.88 obtained by the WordPiece tokenizer. A WordPiece tokenizer with a 64k vocabulary achieved a ratio of 4.12 on the IGC (see Table 6.2), indicating that much of the observed improvement in compression ratio is likely due to the larger vocabulary. Nevertheless, it seems that the Unigram algorithm may be somewhat more effective at capturing subword structure, particularly in morphologically rich languages. Table 6.7 presents the downstream performance of each model, comparing the Unigram tokenizer to a WordPiece tokenizer with a 32k vocabulary.

Overall, the results aligned with our findings for Icelandic. The Unigram tokenizer significantly outperformed WordPiece in eight out of 19 tasks, particularly in NER and text classification (TC), where average scores improved by 0.85 and 0.97 percentage points, respectively. The only case where WordPiece significantly outperformed Unigram was DP for Galician, where it achieved a 0.75 percentage point higher score.

In contrast, a larger vocabulary appeared to have minimal or slightly negative impact on DP and POS, indicating that these tasks may not benefit from reduced subword fragmentation or increased context to the same extent as NER and TC. This suggests that while Unigram tokenization is generally advantageous, its benefits are task-dependent.

Language	Task	WordPiece 32k	Unigram 64k	Δ Score	Rel. Change
Icelandic	POS	97.01%	**97.17%**	0.16%	5.35%
Icelandic	NER	91.48%	**92.13%**	0.65%	7.63%
Icelandic	DP	84.83%	84.84%	0.01%	0.07%
Icelandic	ATS	72.65%	72.60%	-0.05%	-0.18%
Icelandic	QA	60.03%	**60.62%**	0.59%	1.48%
Icelandic	Average	81.20%	81.47%	0.27%	1.45%
Estonian	POS	98.04%	98.01%	-0.03%	-1.53%
Estonian	NER	78.13%	78.63%	0.50%	2.29%
Estonian	DP	89.40%	89.40%	0.00%	0.00%
Estonian	Average	88.52%	88.68%	0.16%	1.37%
Basque	POS	97.00%	97.03%	0.03%	1.00%
Basque	NER	84.42%	84.66%	0.24%	1.54%
Basque	DP	84.45%	84.44%	-0.01%	-0.06%
Basque	Average	88.62%	88.71%	0.09%	0.76%
Galician	POS	98.92%	**99.00%**	0.08%	7.41%
Galician	NER	87.40%	**88.34%**	0.94%	7.46%
Galician	DP	**86.59%**	85.84%	-0.75%	-5.59%
Galician	Average	90.97%	91.06%	0.09%	1.00%
Nepali	POS	96.19%	**96.25%**	0.06%	1.57%
Nepali	NER	90.67%	**91.48%**	0.81%	8.68%
Nepali	TC	78.82%	79.71%	0.89%	4.20%
Nepali	Average	88.56%	89.15%	0.59%	5.13%
Tajik	NER	80.14%	**82.08%**	1.94%	9.77%
Tajik	TC	80.20%	81.25%	1.05%	5.30%
Tajik	Average	80.17%	81.67%	1.49%	7.54%
All	Average	86.12%	86.50%	0.37%	2.70%

Table 6.7: Downstream performance of TEAMS-Small models pre-trained on corpora tokenized using different tokenizer configurations. Scores in **bold** indicate statistically significant differences between models (paired t-test; $p < 0.05$).

6.5 Conclusions

In this chapter, we addressed our third research question: **RQ3: How do different subword tokenization algorithms and vocabulary sizes impact downstream performance for low- and medium-resource languages?** Our experiments across six morphologically rich languages demonstrated that a Unigram tokenizer with a 64k vocabulary consistently outperformed other configurations, particularly for tasks like NER and TC, where scores improved by an average of 0.85 and 0.97 percentage points, respectively. These results align with prior research suggesting that Unigram tokenization better preserves linguistic structure, leading to improved downstream performance (Bostrom and Durrett, 2020), though the magnitude of improvement varied by task and language.

Vocabulary size had a greater impact on downstream performance than the choice of tokenization algorithm, with larger vocabularies consistently yielding better results. However, the benefits were task-dependent, with NER and TC showing substantial improvements, while POS tagging and DP saw minimal gains from increased vocabulary

size. Our evaluation of byte-level tokenization found no significant advantages for high-quality monolingual corpora, indicating that such methods may be more beneficial in multilingual settings, where broader character coverage is essential.

Overall, these findings provide practical guidance for pre-training LMs in resource-constrained settings, highlighting that optimizing vocabulary size can yield greater performance gains than switching between tokenization algorithms.

Chapter 7

Conclusions

The primary goal of this thesis was to investigate strategies for maximizing the performance of Transformer-based language models (LMs) in low- and medium-resource settings. We explored various methods for training LMs under data-constrained conditions.

First, we investigated how best to leverage web-crawled corpora, which are an abundant source of training data, but often suffer from quality issues that can degrade downstream performance. We implemented multiple text filtering techniques and constructed a dataset to evaluate their effectiveness. Based on our observations, we proposed a novel, unsupervised text filtering classifier that is especially well suited for low- and medium-resource languages. We used this classifier to filter noisy web-crawled corpora for six low- to medium-resource languages. We then pre-trained LMs on the filtered and unfiltered corpora and evaluated them on a benchmark of NLP tasks to determine how filtering impacts downstream performance.

Second, we explored multilingual augmentation strategies to optimize performance for a single low- to medium-resource language, based on the hypothesis that linguistic similarity facilitates cross-lingual transfer. To test this, we supplemented monolingual corpora for five diverse languages with text from various sources and evaluated the impact on downstream performance.

Finally, we examined how different subword tokenization algorithms and vocabulary sizes affect downstream performance under low-resource settings.

7.1 Research Questions

The goal of this thesis was to answer three research questions:

RQ1: How do different text filtering techniques impact the downstream performance of LMs pre-trained on web-crawled corpora for low- and medium-resource languages?

In Chapter 4, we explored both rule-based and classifier-based approaches to text quality filtering, evaluating them on TQ-IS, a new dataset we created for Icelandic. Our experiments demonstrated that text filtering significantly impacts the downstream performance of LMs trained on noisy web-crawled corpora. Our results showed that perplexity, computed using an LM trained on a high-quality corpus, is the most effective indicator of document quality, whether used as a rule or a feature in a classifier.

While many heuristic rules are commonly applied for text quality filtering, we found that a small, carefully chosen subset is sufficient to achieve optimal results. Our evaluation of multiple rules showed that only a handful contributed meaningfully to overall classification performance, with three to six rules achieving near-optimal filtering performance. This suggests that many existing rule-based filtering pipelines are overly complex, and that effective filtering can be achieved with a small set of rules with carefully selected threshold values.

We also examined methods for selecting thresholds in heuristic rule-based filtering. Our analysis showed that the Interquartile Range (IQR) method (Nguyen et al., 2023), previously proposed as a simple approach for threshold selection, produces suboptimal results. Instead, we found that a simple visual inspection of the feature distributions provides an intuitive and effective way to determine near-optimal thresholds. By plotting feature pairs, such as perplexity against stop word ratio, we consistently observed a dense cluster of high-quality documents, with low-quality documents appearing as outliers. This pattern was evident across multiple languages, suggesting that visual threshold selection is a generalizable and practical approach.

Based on this observation, we proposed a novel text quality filtering approach using unsupervised clustering and outlier detection algorithms. Our best-performing model in this category, a Gaussian Mixture Model (GMM) classifier, achieved performance comparable to rule-based filtering on TQ-IS, despite requiring no manually labeled training data or extensive feature engineering. While supervised classifiers still outperformed all other methods, clustering-based approaches offer a computationally efficient and language-agnostic alternative for filtering noisy corpora, making them especially valuable for low- and medium-resource languages.

Overall, our findings reinforce the importance of text quality filtering in the pre-training pipeline for LMs trained on web-crawled data. Beyond improving downstream performance, filtering reduces computational costs by discarding noisy data that is likely to be useless or harmful during pre-training. Even when the direct impact on model quality is minimal, the efficiency gains alone make filtering an important step in pre-training, whether training for a fixed number of epochs or a fixed compute budget. Given these benefits, we conclude that effective text quality filtering should be standard practice when pre-training LMs on noisy corpora.

RQ2: How does linguistic similarity influence the effectiveness of cross-lingual transfer in bilingual models?

In Chapter 5, we explored several multilingual augmentation strategies to improve downstream performance for specific low- to medium-resource languages. We evaluated the impact of different types of data, such as text from related languages, unrelated languages, and even programming code, on downstream performance when used to supplement a monolingual corpus. Our findings suggest that linguistic similarity plays a limited role in cross-lingual transfer, particularly for smaller models. Across our experiments, monolingual models generally performed similarly or better than bilingual models, indicating that for small models focused on low-level linguistic tasks, monolingual pre-training remains the most effective approach.

Although small bilingual models showed slightly lower performance than monolingual models, language similarity appeared to mitigate performance degradation in some cases, such as in the Icelandic-Norwegian and Galician-Portuguese models. However, this trend was inconsistent, with the Estonian-Finnish model exhibiting a larger

performance drop than the Basque-Spanish model, despite Estonian and Finnish being closely related while Basque and Spanish are linguistically unrelated. These results suggest that linguistic similarity alone is not a reliable predictor of improved cross-lingual transfer in small bilingual models.

However, our experiments with larger models showed that model capacity plays a crucial role in the effectiveness of bilingual training. In these settings, language similarity had a much stronger impact, with the larger Icelandic-Norwegian model significantly outperforming the Icelandic-Finnish model. This suggests that while smaller models struggle to fully leverage multilingual augmentation, larger models are better equipped to take advantage of linguistic similarities.

Overall, the results suggest that linguistic similarity may contribute to cross-lingual transfer in bilingual models, but its impact depends on model capacity and task type. For smaller models, multilingual augmentation appears to offer little benefit and may even slightly degrade performance. However, for larger models or multilingual tasks, linguistic similarity may offer stronger benefits.

RQ3: How do different subword tokenization algorithms and vocabulary sizes impact downstream performance for low- and medium-resource languages?

In Chapter 6, we investigated how different tokenizer configurations affected downstream performance under resource-constrained settings. Our experiments across six morphologically rich languages demonstrated that a Unigram tokenizer with a 64k vocabulary consistently outperformed other configurations. Notably, we found that increasing vocabulary size had a more significant impact on performance than the choice of tokenization algorithm, suggesting that optimizing vocabulary size should be prioritized when training models for low-resource languages.

The impact of tokenization proved highly task-dependent. Named entity recognition (NER) and token classification (TC) showed substantial improvements, with scores increasing by 0.85 and 0.97 percentage points on average across all languages. These tasks likely benefit from reduced subword fragmentation and increased context length provided by larger vocabularies. In contrast, part-of-speech (POS) tagging and dependency parsing (DP) showed minimal gains or slight performance degradation, indicating that not all tasks benefit equally from optimal tokenizer configurations.

Our evaluation of byte-level tokenization revealed no significant advantages for high-quality monolingual corpora, where unknown tokens are rare. While byte-level approaches ensure complete character coverage, this benefit appears more relevant for multilingual settings where character coverage is crucial than for monolingual models.

These findings provide clear practical guidance for pre-training LMs in resource-constrained settings. While the choice of tokenization algorithm matters, vocabulary size optimization offers more substantial gains in downstream performance.

7.2 Future Work

In future work, we aim to explore additional approaches to text quality filtering, including training a sequence labeling classifier on the TQ-IS dataset to identify low-quality text spans and experimenting with zero-shot classification techniques. We also plan to expand our experiments to include web-crawled corpora from high-resource languages,

such as English, to better understand how effective these filtering techniques are in less resource-constrained settings. Furthermore, we will investigate how the size and diversity of training corpora influence the performance of text quality classifiers. Another key area of research will be analyzing how different types of low-quality text impact the downstream performance of pre-trained LMs. By addressing these questions, we aim to refine existing text quality datasets like TQ-IS and develop robust methodologies for constructing similar datasets for other languages.

For multilingual pre-training, we intend to evaluate the impact of different mixing ratios when augmenting monolingual corpora with donor text. We also hope to experiment with a wider range of model sizes for bilingual corpora to determine whether further increasing model capacity continues to improve the effectiveness of cross-lingual transfer. Finally, we would like to perform a more thorough comparison between multilingual pre-training and other transfer techniques, such as adapter-based methods, lexical adaptation, and knowledge distillation.

Bibliography

Ahmed Abdelali, Hamdy Mubarak, Shammur Chowdhury, Maram Hasanain, Basel Mousi, Sabri Boughorbel, Samir Abdaljalil, Yassine El Kheir, Daniel Izham, Fahim Dalvi, et al. 2024. LAraBench: Benchmarking Arabic AI with Large Language Models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 487–520, St. Julian's, Malta. Association for Computational Linguistics.

David Adelani, Hannah Liu, Xiaoyu Shen, Nikita Vassilyev, Jesujoba Alabi, Yanke Mao, Haonan Gao, and En-Shiun Lee. 2024. SIB-200: A Simple, Inclusive, and Big Evaluation Dataset for Topic Classification in 200+ Languages and Dialects. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 226–245, St. Julian's, Malta. Association for Computational Linguistics.

Itziar Aduriz, Maria Jesus Aranzabe, Jose Maria Arriola, Aitziber Atutxa, A Díaz De Ilarraza, Aitzpea Garmendia, and Maite Oronoz. 2003. Construction of a Basque Dependency Treebank. In *Proceedings of the Second Workshop on Treebanks and Linguistic Theories (TLT)*, pages 201–204, Växjö, Sweden.

Abien Fred Agarap. 2019. Deep Learning using Rectified Linear Units (ReLU). *arXiv preprint arXiv:1803.08375*.

Rodrigo Agerri, Nú Bel, German Rigau, and Horacio Saggion. 2018a. TUNER: Multifaceted Domain Adaptation for Advanced Textual Semantic Processing. First Results Available. *Procesamiento del Lenguaje Natural*, 61:163–166. Copyright - Copyright Sociedad Española para el Procesamiento del Lenguaje Natural Sep 2018; Last updated - 2024-08-27.

Rodrigo Agerri, Xavier Gómez Guinovart, German Rigau, and Miguel Anxo Solla Portela. 2018b. Developing New Linguistic Resources and Tools for the Galician Language. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Rodrigo Agerri, Iñaki San Vicente, Jon Ander Campos, Ander Barrena, Xabier Saralegi, Aitor Soroa, and Eneko Agirre. 2020. Give your Text Representation Models some Love: the Case for Basque. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4781–4788, Marseille, France. European Language Resources Association.

Kabir Ahuja, Harshita Diddee, Rishav Hada, Millicent Ochieng, Krithika Ramesh, Prachi Jain, Akshay Nambi, Tanuja Ganu, Sameer Segal, Mohamed Ahmed, et al.

2023. MEGA: Multilingual Evaluation of Generative AI. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4232–4267, Singapore. Association for Computational Linguistics.

Iñaki Alegria and Kepa Sarasola. 2017. Language Technology for Language Communities: An Overview based on Our Experience. In *Communities in Control: Learning tools and strategies for multilingual endangered language communities", CinC*, pages 19–21.

Iñaki Alegria, Olatz Arregi, Irene Balza, Nerea Ezeiza, Izaskun Fernandez, and Ruben Urizar. 2004. Design and Development of a Named Entity Recognizer for an Agglutinative Language. In *First International Joint Conference on NLP (IJCNLP-04). Workshop on Named Entity Recognition.*

Mehdi Ali, Michael Fromm, Klaudia Thellmann, Richard Rutmann, Max Lübbering, Johannes Leveling, Katrin Klug, Jan Ebert, Niclas Doll, Jasper Buschhoff, et al. 2024. Tokenizer Choice For LLM Training: Negligible or Crucial? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3907–3924, Mexico City, Mexico. Association for Computational Linguistics.

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. 2023. PaLM 2 Technical Report. *arXiv preprint arXiv:2305.10403.*

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, et al. 2024. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. In *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*. ACM.

Maria Jesús Aranzabe, Aitziber Atutxa, Kepa Bengoetxea, Arantza Diaz de Ilarraza, Iakes Goenaga, Koldo Gojenola, and Larraitz Uria. 2015. Automatic Conversion of the Basque Dependency Treebank to Universal Dependencies. In *Proceedings of the Fourteenth International Workshop on Treebanks and Linguistic Theories (TLT14)*, pages 233–241, Warszawa, Poland. Institute of Computer Science of the Polish Academy of Sciences.

Þórunn Arnardóttir, Hinrik Hafsteinsson, Einar Freyr Sigurðsson, Kristín Bjarnadóttir, Anton Karl Ingason, Hildur Jónsdóttir, and Steinþór Steingrímsson. 2020. A Universal Dependencies Conversion Pipeline for a Penn-format Constituency Treebank. In *Proceedings of the Fourth Workshop on Universal Dependencies (UDW 2020)*, pages 16–25, Barcelona, Spain (Online). Association for Computational Linguistics.

Mikel Artetxe, Itziar Aldabe, Rodrigo Agerri, Olatz Perez-de Viñaspre, and Aitor Soroa. 2022. Does Corpus Quality Really Matter for Low-Resource Languages? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7383–7390, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the Cross-lingual Transferability of Monolingual Representations. In *Proceedings of the 58th Annual*

Meeting of the Association for Computational Linguistics, pages 4623–4637, Online. Association for Computational Linguistics.

Giuseppe Attardi, Daniele Sartiano, and Maria Simi. 2021. Biaffine Dependency and Semantic Graph Parsing for Enhanced Universal Dependencies. In *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, pages 184–188, Online. Association for Computational Linguistics.

Marta Bañón, Miquel Esplà-Gomis, Mikel L. Forcada, Cristian García-Romero, Taja Kuzman, Nikola Ljubešić, Rik van Noord, Leopoldo Pla Sempere, Gema Ramírez-Sánchez, Peter Rupnik, et al. 2022. MaCoCu: Massive collection and curation of monolingual and bilingual data: focus on under-resourced languages. In *Proceedings of the 23rd Annual Conference of the European Association for Machine Translation*, pages 303–304, Ghent, Belgium. European Association for Machine Translation.

Adrien Barbaresi. 2021. Trafilatura: A Web Scraping Library and Command-Line Tool for Text Discovery and Extraction. 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*, pages 122–131, Online. Association for Computational Linguistics.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media, Inc.

Kristín Bjarnadóttir, Kristín Ingibjörg Hlynsdóttir, and Steinþór Steingrímsson. 2019. DIM: The Database of Icelandic Morphology. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 146–154, Turku, Finland. Linköping University Electronic Press.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Kaj Bostrom and Greg Durrett. 2020. Byte Pair Encoding is Suboptimal for Language Model Pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, Online. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language Models are Few-Shot Learners. *arXiv preprint arXiv:2005.14165*.

Martin Juan José Bucher and Marco Martini. 2024. Fine-Tuned 'Small' LLMs (Still) Significantly Outperform Zero-Shot Generative AI Models in Text Classification. *arXiv preprint arXiv:2406.08660*.

Jasone Cenoz. 2000. Basque, Spanish, French and English in the Basque Country. Paper presented at the Seminar on Comparative Perspectives in Multicultural Europe, Oegstgeest, Netherlands.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating Long Sequences with Sparse Transformers. *arXiv preprint arXiv:1904.10509*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv preprint arXiv:1406.1078*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *8th International Conference on Learning Representations, ICLR 2020*, Addis Ababa, Ethiopia.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual Language Model Pretraining. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating Cross-lingual Sentence Representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Jón Daðason and Hrafn Loftsson. 2024. Text Filtering Classifiers for Medium-Resource Languages. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 15789–15801, Torino, Italia. ELRA and ICCL.

Jón Daðason, Hrafn Loftsson, Salome Sigurðardóttir, and Þorsteinn Björnsson. 2021. IceSum: An Icelandic Text Summarization Corpus. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 9–14, Online. Association for Computational Linguistics.

Jón Friðrik Daðason and Hrafn Loftsson. 2022. Pre-training and Evaluating Transformer-based Language Models for Icelandic. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 7386–7391, Marseille, France. European Language Resources Association.

Iria de Dios-Flores, Carmen Magariños, Adina Ioana Vladu, John E. Ortega, José Ramom Pichel, Marcos García, Pablo Gamallo, Elisa Fernández Rei, Alberto Bugarín-Diz, Manuel González González, Senén Barro, and Xosé Luis Regueira. 2022. The Nós Project: Opening routes for the Galician language in the field of

language technologies. In *Proceedings of the Workshop Towards Digital Language Equality within the 13th Language Resources and Evaluation Conference*, pages 52–61, Marseille, France. European Language Resources Association.

Ona de Gibert, Graeme Nail, Nikolay Arefyev, Marta Bañón, Jelmer van der Linde, Shaoxiong Ji, Jaume Zaragoza-Bernabeu, Mikko Aulamo, Gema Ramírez-Sánchez, Andrey Kutuzov, Sampo Pyysalo, Stephan Oepen, and Jörg Tiedemann. 2024. A New Massive Multilingual Dataset for High-Performance Language Technologies. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 1116–1128, Torino, Italia. ELRA and ICCL.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. BERTje: A Dutch BERT Model. *arXiv preprint arXiv:1912.09582*.

DeepSeek-AI. 2024. DeepSeek-V3 Technical Report. *arXiv preprint arXiv:2412.19437*.

Frederik Michel Dekking, Cornelis Kraaikamp, Hendrik Paul Lopuhaä, and Ludolf Erwin Meester. 2005. *A Modern Introduction to Probability and Statistics: Understanding Why and How*, 1 edition. Springer Texts in Statistics. Springer London.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Philip Gage. 1994. A New Algorithm for Data Compression. *The C Users Journal*, 12(2):23–38.

Marcos Garcia. 2024. Training and evaluation of vector models for Galician. *Language Resources and Evaluation*, 58(4):1419–1462.

Marcos Garcia, Carlos Gómez-Rodríguez, and Miguel A. Alonso. 2018. New Treebank or Repurposed? On the Feasibility of Cross-Lingual Parsing of Romance languages with Universal Dependencies. *Natural Language Engineering*, 24(1):91–122.

Charlotte Gooskens and Hanna-Ilona Härmävaara. 2019. Mutual intelligibility of Finnish and Estonian vocabulary. *Lähivõrdlusi. Lähivertailuja*, 29:13–56.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*.

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018. Meta-Learning for Low-Resource Neural Machine Translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631, Brussels, Belgium. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. *arXiv preprint arXiv:2006.03654*.

Kenneth Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre Is Not Superb: Derivational Morphology Improves BERT's Interpretation of Complex Words. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3594–3608, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Svanhvít L. Ingólfsdóttir, Ásmundur A. Guðjónsson, and Hrafn Loftsson. 2020. Named Entity Recognition for Icelandic: Annotated Corpus and Models. In *Proceedings of the 8th International Conference on Statistical Language and Speech Processing (SLSP 2020)*, pages 46–57, Cardiff, United Kingdom.

Zi-Hang Jiang, Weihao Yu, Daquan Zhou, Yunpeng Chen, Jiashi Feng, and Shuicheng Yan. 2020. ConvBERT: Improving BERT with Span-based Dynamic Convolution. In *Advances in Neural Information Processing Systems*, volume 33, pages 12837–12848. Curran Associates, Inc.

Yiğit Bekir Kaya and A. Cüneyd Tantuğ. 2024. Effect of tokenization granularity for Turkish large language models. *Intelligent Systems with Applications*, 21:200335.

R. Kneser and H. Ney. 1995. Improved backing-off for M-gram language modeling. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 181–184 vol.1.

Denis Kocetkov, Raymond Li, Loubna Ben allal, Jia Li, Chenghao Mou, Yacine Jernite, Margaret Mitchell, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, et al. 2023. The Stack: 3 TB of permissively licensed source code. *Transactions on Machine Learning Research*.

Kristina Koppel and Jelena Kallas. 2022. Eesti keele ühendkorpuste sari 2013–2021: mahukaim eestikeelsete digitekstide kogu. *Eesti Rakenduslingvistika Ühingu aastaraamat*, 18:207–228.

Julia Kreutzer, Isaac Caswell, Lisa Wang, Ahsan Wahab, Daan van Esch, Nasanbayar Ulzii-Orshikh, Allahsera Tapo, Nishant Subramani, Artem Sokolov, Claytone

Sikasote, et al. 2022. Quality at a Glance: An Audit of Web-Crawled Multilingual Datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72.

Taku Kudo. 2018. 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)*, pages 66–75, Melbourne, Australia.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *arXiv preprint arXiv:1909.11942*.

Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada, Huu Nguyen, et al. 2022. The BigScience ROOTS Corpus: A 1.6TB Composite Multilingual Dataset. In *Advances in Neural Information Processing Systems*, volume 35, pages 31809–31826. Curran Associates, Inc.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation Forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422.

Yang Liu and Mirella Lapata. 2019. Text Summarization with Pretrained Encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Hrafn Loftsson, Jökull H. Yngvason, Sigrún Helgadóttir, and Eiríkur Rögnvaldsson. 2010. Developing a PoS-tagged corpus using existing tools. In *Proceedings of 7th SaLTMiL Workshop on Creation and Use of Basic Lexical Resources for Less-Resourced Languages*, LREC 2010, Valetta, Malta.

Marco Lui and Timothy Baldwin. 2012. langid.py: An Off-the-shelf Language Identification Tool. In *Proceedings of the ACL 2012 System Demonstrations*, pages 25–30, Jeju Island, Korea. Association for Computational Linguistics.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric de la Clergerie, Djamé Seddah, and Benoît Sagot. 2020. CamemBERT: a Tasty French Language Model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7203–7219, Online. Association for Computational Linguistics.

Mary L. McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica*, 22(3):276–282.

Meryem M'hamdi, Xiang Ren, and Jonathan May. 2023. Cross-lingual Continual Learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3908–3943, Toronto, Canada. Association for Computational Linguistics.

Vincent Micheli, Martin d'Hoffschmidt, and François Fleuret. 2020. On the importance of pre-training data volume for compact language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7853–7858, Online. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, Lake Tahoe, Nevada.

Anthony Moi and Nicolas Patry. 2023. HuggingFace's Tokenizers.

Henrique Monteagudo. 2024. Commentary: Language Policy in Galicia, 1980-2020. An Overview. *Journal on Ethnopolitics and Minority Issues in Europe*, 23(1):1–20.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. 2023. Scaling Data-Constrained Language Models. *arXiv preprint arXiv:2305.16264*.

Kadri Muischnek, Kaili Müürisep, and Tiina Puolakainen. 2016. Estonian Dependency Treebank: from Constraint Grammar tagset to Universal Dependencies. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1558–1565, Portorož, Slovenia. European Language Resources Association (ELRA).

Kadri Muischnek, Kaili Müürisep, Tiina Puolakainen, Eleri Aedmaa, Riin Kirt, and Dage Särg. 2014. Estonian Dependency Treebank and its annotation scheme. In *Proceedings of the Thirteenth International Workshop on Treebanks and Linguistic Theories (TLT13)*, pages 285–291.

Thuat Nguyen, Chien Van Nguyen, Viet Dac Lai, Hieu Man, Nghia Trung Ngo, Franck Dernoncourt, Ryan A. Rossi, and Thien Huu Nguyen. 2023. CulturaX: A Cleaned, Enormous, and Multilingual Dataset for Large Language Models in 167 Languages. *arXiv preprint arXiv:2309.09400*.

Anna Nikulásdóttir, Jón Guðnason, Anton Karl Ingason, Hrafn Loftsson, Eiríkur Rögnvaldsson, Einar Freyr Sigurðsson, and Steinþór Steingrímsson. 2020. Language Technology Programme for Icelandic 2019-2023. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 3414–3422, Marseille, France. European Language Resources Association.

Anna Björk Nikulásdóttir, Þórunn Arnardóttir, Jón Guðnason, Þorsteinn Daði Gunnarsson, Anton Karl Ingason, Haukur Páll Jónsson, Hrafn Loftsson, Hulda Óladóttir, Einar Freyr Sigurðsson, Atli Þór Sigurgeirsson, Vésteinn Snæbjarnarson, and Steinþór Steingrímsson. 2022. Help Yourself from the Buffet: National Language

Technology Infrastructure Initiative on CLARIN-IS. In *Selected Papers from the CLARIN Annual Conference 2021*, pages 109–125.

Nobal Niraula and Jeevan Chapagain. 2022. Named Entity Recognition for Nepali: Data Sets and Algorithms. *The International FLAIRS Conference Proceedings*, 35.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, et al. 2022. No Language Left Behind: Scaling Human-Centered Machine Translation. *arXiv preprint arXiv:2207.04672*.

OpenAI. 2024. GPT-4 Technical Report. *arXiv preprint arXiv:2302.10198*.

Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2020. A Monolingual Approach to Contextualized Word Embeddings for Mid-Resource Languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual Name Tagging and Linking for 282 Languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale. *arXiv preprint arXiv:2406.17557*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Jan Pomikálek. 2011. *Removing Boilerplate and Duplicate Content from Web Corpora*. Ph.D. thesis, Masaryk University, Faculty of Informatics, Brno, Czech Republic.

Sampo Pyysalo, Jenna Kanerva, Antti Virtanen, and Filip Ginter. 2021. WikiBERT Models: Deep Transfer Learning for Many Languages. In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 1–10, Reykjavik, Iceland (Online). Linköping University Electronic Press, Sweden.

Faisal Qarah and Tawfeeq Alsanoosy. 2024. A Comprehensive Analysis of Various Tokenizers for Arabic Large Language Models. *Applied Sciences*, 14(13).

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-Training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2022. Scaling Language Models: Methods, Analysis & Insights from Training Gopher. *arXiv preprint arXiv:2112.11446*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Afshin Rahimi, Yuan Li, and Trevor Cohn. 2019. Massively Multilingual Transfer for NER. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 151–164, Florence, Italy. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020. Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Leonard Richardson. 2020. Beautiful soup. Version 4.9.3.

Nathaniel Robinson, Perez Ogayo, David R. Mortensen, and Graham Neubig. 2023. ChatGPT MT: Competitive for High- (but Not Low-) Resource Languages. In *Proceedings of the Eighth Conference on Machine Translation*, pages 392–418, Singapore. Association for Computational Linguistics.

Eiríkur Rögnvaldsson, Anton Karl Ingason, Einar Freyr Sigurðsson, and Joel Wallenberg. 2012. The Icelandic Parsed Historical Corpus (IcePaHC). In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 1977–1984, Istanbul, Turkey. European Language Resources Association (ELRA).

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2023. BLOOM: A 176B-Parameter Open-Access Multilingual Language Model. *arXiv preprint arXiv:2211.05100*.

Stefan Schweter. 2020. BERTurk - BERT models for Turkish.

Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. 2001. Estimating the Support of a High-Dimensional Distribution. *Neural Computation*, 13(7):1443–1471.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. 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)*, pages 1715–1725, Berlin, Germany.

Noam Shazeer. 2020. GLU Variants Improve Transformer. *arXiv preprint arXiv:2002.05202*.

Jiaming Shen, Jialu Liu, Tianqi Liu, Cong Yu, and Jiawei Han. 2021. Training ELECTRA Augmented with Multi-word Selection. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2475–2486, Online. Association for Computational Linguistics.

Kairit Sirts. 2023. Estonian Named Entity Recognition: New Datasets and Models. In *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 752–761, Tórshavn, Faroe Islands. University of Tartu Library.

Njáll Skarphéðinsson, Breki Guðmundsson, Steinar Þ. Smári, Marta K. Lárusdóttir, Hafsteinn Einarsson, Abuzar Khan, Eric Nyberg, and Hrafn Loftsson. 2023. GameQA: Gamified Mobile App Platform for Building Multiple-Domain Question-Answering Datasets. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 152–160, Dubrovnik, Croatia. Association for Computational Linguistics.

Vésteinn Snæbjarnarson, Annika Simonsen, Goran Glavaš, and Ivan Vulić. 2023. Transfer to a Low-Resource Language via Close Relatives: The Case Study on Faroese. In *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 728–737, Tórshavn, Faroe Islands. University of Tartu Library.

Vésteinn Snæbjarnarson. 2021. Automated methods for Question-Answering in Icelandic. Master's thesis, University of Iceland.

Steinþór Steingrímsson, Sigrún Helgadóttir, Eiríkur Rögnvaldsson, Starkaður Barkarson, and Jón Guðnason. 2018. Risamálheild: A Very Large Icelandic Text Corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Steinþór Steingrímsson, Hrafn Loftsson, and Andy Way. 2023. Filtering Matters: Experiments in Filtering Training Sets for Machine Translation. In *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 588–600, Tórshavn, Faroe Islands. University of Tartu Library.

Cagri Toraman, Eyup Halit Yilmaz, Furkan Şahinuç, and Oguzhan Ozcelik. 2023. Impact of Tokenization on Language Models: An Analysis for Turkish. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 22(4).

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288*.

Rik van Noord, Taja Kuzman, Peter Rupnik, Nikola Ljubešić, Miquel Esplà-Gomis, Gema Ramírez-Sánchez, and Antonio Toral. 2024. Do Language Models Care About Text Quality? Evaluating Web-Crawled Corpora Across 11 Languages. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 5221–5234, Torino, Italia. ELRA and ICCL.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Kadri Vider, Krista Liin, and Neeme Kahusk. 2012. Strategic Importance of Language Technology in Estonia. In *Human Language Technologies — The Baltic Perspective*. IOS Press.

Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. 2019. Multilingual is not enough: BERT for Finnish. *arXiv preprint arXiv:1912.07076*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, et al. 2024. Smarter, Better, Faster, Longer: A Modern Bidirectional Encoder for Fast, Memory Efficient, and Long Context Finetuning and Inference. *arXiv preprint arXiv:2412.13663*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2024. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. CCNet: Extracting High Quality Monolingual Datasets from Web Crawl Data. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France. European Language Resources Association.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Shaohua Wu, Xudong Zhao, Tong Yu, Rongguo Zhang, Chong Shen, Hongli Liu, Feng Li, Hong Zhu, Jiangang Luo, Liang Xu, and Xuanwei Zhang. 2021. Yuan 1.0: Large-Scale Pre-trained Language Model in Zero-Shot and Few-Shot Learning. *arXiv preprint arXiv:2110.04725*.

Shijie Wu and Mark Dredze. 2020. Are All Languages Created Equal in Multilingual BERT? In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 120–130, Online. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv preprint arXiv:1609.08144*.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex Machina: Personal Attacks Seen at Scale. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, page 1391–1399, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Yogendra P. Yadava, Andrew Hardie, Ram Raj Lohani, Bhim N. Regmi, Srishtee Gurung, Amar Gurung, Tony McEnery, Jens Allwood, and Pat Hall. 2008. Construction and annotation of a corpus of contemporary Nepali. *Corpora*, 3(2):213–225.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, et al. 2024. Yi: Open Foundation Models by 01.AI. *arXiv preprint arXiv:2403.04652*.

Hao Yu, Zachary Yang, Kellin Pelrine, Jean Francois Godbout, and Reihaneh Rabbany. 2023. Open, Closed, or Small Language Models for Text Classification? *arXiv preprint arXiv:2308.10092*.

Hongkun Yu, Chen Chen, Xianzhi Du, Yeqing Li, Abdullah Rashwan, Le Hou, Pengchong Jin, Fan Yang, Frederick Liu, Jaeyoun Kim, and Jing Li. 2020. TensorFlow Model Garden.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Shiyue Zhang, Vishrav Chaudhary, Naman Goyal, James Cross, Guillaume Wenzek, Mohit Bansal, and Francisco Guzman. 2022. How Robust is Neural Machine Translation to Language Imbalance in Multilingual Tokenizer Training? In *Proceedings of the 15th biennial conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 97–116, Orlando, USA. Association for Machine Translation in the Americas.

Qihuang Zhong, Liang Ding, Juhua Liu, Bo Du, and Dacheng Tao. 2023. Can ChatGPT Understand Too? A Comparative Study on ChatGPT and Fine-tuned BERT. *arXiv preprint arXiv:2302.10198*.

Zhi-Hua Zhou. 2017. A brief introduction to weakly supervised learning. *National Science Review*, 5(1):44–53.

Joey Öhman, Severine Verlinden, Ariel Ekgren, Amaru Cuba Gyllensten, Tim Isbister, Evangelia Gogoulou, Fredrik Carlsson, and Magnus Sahlgren. 2023. The Nordic Pile: A 1.2TB Nordic Dataset for Language Modeling. *arXiv preprint arXiv:2303.17183*.