Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

SlovakBabyLM: Improving Language Models Through Cognitive and Linguistic Principles in Slovak language

Diploma thesis

2024
bc. Ľuboš Kriš

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

SlovakBabyLM: Improving Language Models Through Cognitive and Linguistic Principles in Slovak language

Diploma thesis

Study program:	Cogniive Science
Department:	Department of Applied Informatics
Supervisor:	Mgr. Marek Šuppa

Bratislava, 2024
bc. Ľuboš Kriš

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Ľuboš Kriš
Študijný program: kognitívna veda (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: anglický
Sekundárny jazyk: slovenský

Názov: SlovakBabyLM: Improving Language Models Through Cognitive and Linguistic Principles in Slovak language
SlovakBabyLM: Zlepšovanie jazykových modelov prostredníctvom kognitívnych a lingvistických princípov v slovenskom jazyku

Anotácia: Učenie založené na učebných osnovách (curriculum learning) je špecifická tréningová stratégia pre jazykové modely. V anglicky orientovanom výskume spracovania prirodzeného jazyka existuje dataset a súťaž, ktoré vedú k optimalizácii tréningu modelov a ich trénovaniu na objeme dát približujúcom sa ľudskej jazykovej expozícii. Podľa dostupných informácií podobný dataset v slovenčine zatiaľ neexistuje. Vytvorenie takéhoto datasetu by mohlo podnietiť ďalší výskum v oblasti učenia založeného na učebných osnovách a spracovania slovenského jazyka pomocou jazykových modelov.

Cieľ: Ciele tejto práce zahŕňajú, avšak nie sú limitované na, nasledovné:
1. Preskúmať literatúru k kurikulárnemu učeniu (curriculum learning) a procesom osvojovania jazyka.
2. Zostaviť slovenský korpus napodobňujúci detskú slovnú zásobu podľa kritérií BabyLM Challenge.
3. Navrhnúť, implementovať a experimentálne vyhodnotiť kurikulárne stratégie pre jazykové modelovanie v slovenčine.

Literatúra:
Wang, X., Chen, Y., and Zhu, W. (2021). A survey on curriculum learning. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(9): 4555–4576.
Warstadt, A. a spol. (2023). Call for papers – the BabyLM challenge: Sample-efficient pretraining on a developmentally plausible corpus. arXiv:2301.11796.

Vedúci: Mgr. Marek Šuppa
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: doc. RNDr. Tatiana Jajcayová, PhD.

Dátum zadania: 19.05.2025

Dátum schválenia: 19.03.2024

prof. Ing. Igor Farkaš, Dr.
garant študijného programu

.....
študent

.....
vedúci práce

Comenius University Bratislava
Faculty of Mathematics, Physics and Informatics

THESIS ASSIGNMENT

Name and Surname:	Ľuboš Kriš
Study programme:	Cognitive Science (Single degree study, master II. deg., full time form)
Field of Study:	Computer Science
Type of Thesis:	Diploma Thesis
Language of Thesis:	English
Secondary language:	Slovak

Title: SlovakBabyLM: Improving Language Models Through Cognitive and Linguistic Principles in Slovak language

Annotation: Curriculum learning is a specific training strategy for language models based on instructional curricula. In English-focused NLP research, there exists a dataset and competition that drive optimization of model training on a volume of data approaching human language exposure. According to available information, no comparable Slovak dataset yet exists. Creating such a dataset could stimulate further research into curriculum learning and the processing of the Slovak language with language models.

Aim: The objectives of this work include, but are not limited to, the following:
1. Survey the literature on curriculum learning and the process of language acquisition.
2. Compile a Slovak corpus emulating child#directed vocabulary in accordance with BabyLM Challenge criteria.
3. Design, implement, and empirically evaluate curriculum strategies for Slovak language modeling.

Literature: Wang, X., Chen, Y., and Zhu, W. (2021). A survey on curriculum learning. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(9): 4555–4576.
Warstadt, A., Choshen, L., Mueller, A., Williams, A., Wilcox, E., and Zhuang, C. (2023). Call for papers – the BabyLM challenge: Sample-efficient pretraining on a developmentally plausible corpus. arXiv:2301.11796.

Supervisor:	Mgr. Marek Šuppa
Department:	FMFI.KAI - Department of Applied Informatics
Head of department:	doc. RNDr. Tatiana Jajcayová, PhD.
Assigned:	19.05.2025

Approved: 19.03.2024 prof. Ing. Igor Farkaš, Dr.
Guarantor of Study Programme

Student Supervisor

Acknowledgment: I would like to express my gratitude to Mgr. Marek Šuppa for his academic support and for providing the technical means to carry out this work.

Abstrakt

V súčasnosti môžeme pozorovať trend v oblasti spracovania prirodzeného jazyka, v neustálom vytváraní nových, väčších a čo raz lepších jazykových modelov v rôznych oblastiach spracovania prirodzeného jazyka. Nielen čo sa týka architektúry modela ale aj dát, vdaka ktorým acquire language. Avšak pre tieto potreby sú potrebné určité materiálne zdroje, čo môže byť náročné. Špeciálne v jazykoch s nízkymi zdrojmi dát, čo môže sťažovať vývin v oblasti spracovania prirodzeného jazyka pre špecifické jazyky. Preto sme sa inspširovali ľudským vývinom jazyka a vytvorili SlovakBabylm dataset, a následne otestovali rôzne metódy na optimalizáciu tréningu jazykového modelu, ktoré sa používajú v Angličtine na Slovenský jazyk.

Kľúčové slová: pred-trénovanie jazykových modelov, spracovanie prirodzeného jazyka, osvojenie jazyka

Abstract

Contemporary, we can observe a trend in the field of natural language processing, in the constant creation of new, bigger and better language models in different areas of natural language processing. Not only in terms of the architecture of the model but also in terms of the data thanks to which we acquire language. However, for these needs, certain material resources are required, which can be challenging. Especially in languages with low data resources, which can make it difficult to develop in the area of natural language processing for specific languages. Therefore, we were inspired by human language evolution and created the SlovakBabylm dataset, and then tested different methods to optimize the training of the language model used in English to Slovak.

Keywords: pre-training language models, natural language processing, language acquisition

Contents

List of Figures

List of Tables

List of Abbreviations

ANN Artificial Neural Network.

BCL Balanced Curriculum Learning.

BERT Bidirectional Encoder Representations from Transformers.

CL Curriculum Learning.

DL Deep Learning.

FFN Feed-Forward Neural Network.

LLM Large Language Models.

LM Language Model.

LRL Low-Resource Languages.

ML Machine Learning.

MLM Multilingual Language Model.

NLP Natural Language Processing.

PCL Progressive Curriculum Learning.

Introduction

In recent years, we can observe a trend of creating various specific Language Model (LM) within the Slavic language family, such as Czert (Sido et al., 2021) HerBERT (Mroczkowski et al., 2021) SlovakBERT (Pikuliak et al., 2021), which use the Bidirectional Encoder Representations from Transformers (BERT) architecture to create an LM in the specific languages. These models are among the first in the framework of these languages. However, in the framework of these models, we do not use knowledge consisting of psycholinguistics, and we do not focus on the cognitive plausibility of a given model. The relevance lies in the development of LM to the human way of language production, which can be seen as a step forward to a better understanding of the LM by humans, and thus the user of the model would better predict the outcome and quality of the results of the language model and avoid potential sources of errors Beinborn and Hollenstein (2023). Additionally, from a materialistic perspective. Creating LM can be computationally exhaustive and in languages with a small amount of text is problematic to source enough text to create LM. The application of linguistic metrics can help reduce the amount of text needed to develop a LM.

Therefore, we try to replicate BabyLM Challenge (Warstadt et al., 2023a) in the Slovak language. This Challenge created a dataset resembling human speech, which could fulfill the properties of the human resource to be acquired, such as the gradual acquisition of more complex words or the amount of acquired vocabulary. The BabyLM challenge provides an opportunity for researchers to test the training of language models. However, we cannot find such a dataset in the Slovak language. From the individual models, the problem of creating a sufficiently large dataset in the model for pre-training arises. Therefore, the goal is to create a dataset in the Slovak language using the BabyLM challenge. At the same time, we try to create a model based on cognitive plausibility and a model where we apply the our created dataset. We apply and create metrics stem from the BabyLM challenge.

The first chapter is about the foundation of each discipline and the methods that can be used to achieve our goals. Also, we mention reasons why we created Slovak datasets and why reasearch in Slovak language is important. In the second chapter, Base methods are mentioned, how text was gained from sources, which methods we used for it. How this text was used and how our metrics manipulate with text. In final third chapter, we provide results of our solutions and provide analysis of our metrics.

Chapter 1

Literature and Related work

1.1 Natural language processing

Natural Language Processing (NLP) is a subfield of artificial intelligence focused on processing, understanding, interpreting, and generating human language. NLP stands at the intersection of various intellectual disciplines. NLP uses theoretical linguistics as a basis for the creation and analysis of computational algorithms and natural language representations. We rely on computational linguistics and Machine Learning (ML) to create and analyze NLP computational algorithms. Furthermore, we can mention Computer Science or Speech Processing and Ethics and others that are involved in the creation of NLP techniques. Computational Linguistics combines linguistics and computer science on the basis of creating computational methods from basic linguistic knowledge and one of the many goals is for example extracting information from texts, translating text, understanding text or receiving instructions using language and others. ML and especially NLP offers a wide range of general techniques for a variety of tasks, including translating a sequence of discrete tokens from one dictionary into a sequence of discrete tokens in another dictionary (Eisenstein, 2018). NLP tasks can be divided into text analysis and text generation, where we can create tools for text generation thanks to text analysis. The individual parts have their own specifics, so for research purposes we will discuss text analysis in more detail. Text analysis can take place from different perspectives such as syntactic, semantic, lexical, grammatical or discourse. Syntax analysis focuses on analyzing the structure of sentences in a text and regulating this structure so that the content of the sentence is preserved. Grammatical analysis focuses on the grammatical rules found in a particular language. Semantic analysis focuses on the semantic meaning of sentences and discourse analysis focuses on analyzing the connections and contexts of sentences into a coherent whole (Chowdhary, 2020).

Initiation of the endeavor in NLP stems from the human need to quickly translate sentences from different languages. In particular, during the Second World War, scientists attempted to create machine translation between English and Russian (Jones, 1994). From the grammatical point of view, we can mention Noam Chomsky and his contribution to Universal Grammar, which will be advanced in Curriculum learning part. (Chomsky, 1980) Initial scientific research in the NLP field was focused on rules. Example of this trend is computer program Eliza (Weizenbaum, 1966). Eliza work with decomposition rules of text, which takes important words from a sentence. Output was created by reassembly rules, which were connected with the input rules. It also had the rules of basic English and the relationships between words. For example, the word you should be followed by the word are. Around 1990s to 2000s start connecting Deep Learning (DL) and machine learning (ML) to NLP and scientific research made shift from rule-based NLP to implement statistical and neural network-based methods. We can mention proposal of the Feed-Forward Neural Network (FFN), which predicts next word (Bengio et al., 2003) and methods such as Support vector machine (Cortes and Vapnik, 1995) in the field of ML, which help with classification task. From commercial point of view, Google introduce Google Translate based on Statistical Machine Translation (James, 2023). Continuesly innovations like Word2Vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014), which focus on capture deeper semantic relationships between words using multidimensional vectors. But all previous innovations lead to the creation of Transformer architecture (Vaswani, 2017), which can be considered as a foundation stone of current research in NLP. from which current language models such as GPT (Radford and Narasimhan, 2018) or Bert (Devlin et al., 2019) draw.

1.2 Transfomers

Transformer is a type of DL model for natural language, captures contextual relationships without the need for recurrent or convolutional structures, introduced by Vaswani et al. (2017) in the paper "Attention is All You Need". Now we roughly describe how the Transformer architecture works, and in the following subsections we will discuss the individual functions and processes in more detail. The main parts of the Transformer architecture are ANN and attention mechanisms. The input to the Transformers is text, which is transformed into tokens, which represent part of text from the transformer vocabulary. The given text is divided into tokens. Then the token is transformed into a numerical representation of the token. Transformers utilize learned embeddings from previously gained language through weights in the attention mechanism and neural networks. Thanks, this mechanism Transformers, is able to convert the input tokens to embeddings, which captures meaning and context of text. Additionally, is incorpo-

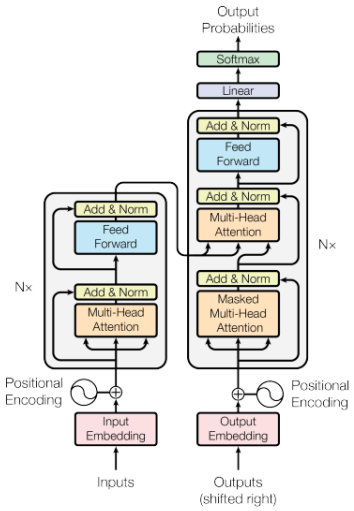

Figure 1.1: Transformer architecture Vaswani et al. (2017)

rated positional embedding. The positions of given words in a sentence are added due to transformers no having recurrence or convolution, which process text sequentially or spatially, thus the order of words in the sentence is learned. The created embeddings are passed through an attention mechanism that highlights important areas of text for the neural network. The created embeddings are passed through an attention mechanism that highlights important areas of text for the feed-forward network. The given processes are wrapped into two parts, namely encoder and decoder. The encoder processes the input sequence, and the decoder generates the output sequence. Each layer of the encoder and decoder consists of several sub-layers of feed-forward neural networks and several attention heads. Thanks to auto-regressivity, it can have the predictions of being better than when it consumes previously generated symbols as additional input when generating the next one because each word is mapped to a vector that the model learns during training. The process of creating a Transformer model involves training, where the Transformers retrieves statistical data on individual occurrences of words and phrases in a specific context, and then the Transformer is fine-tuned to a specific method (Vaswani et al., 2017)

1.2.1 Artificial neural network (ANN)

The connectionist paradigm explains that through the activity of neurons in our brain, we form our mental representations. Individual neurons (or small bundles of neurons) are devoted to the representation of a phenomenon or object that the brain needs to record. (Buckner and Garson, 2025) Therefore, we can observe the inspiration of the ANN in its architecture. The ANN is made up of a number of perceptrons stacked in layers.

Perceptron has 3 parts. The first part accepts signals from raw input or other perceptrons. The second part sums all positive and negative stimuli into one stimulus and last part decide if this stimulus is powerful enough. ANN utilizes vectors as a representation of stimulus; additionally has biases and weights, which influence the acceptance of certain stimuli. Stimulus reception and firing of a neuron depends on transfer function or in other words activation function. There are many transfer functions for example Step function, the Linear function or (Sigmoid) function. Each accepts a number as input, the processing of which is based on a mathematical function that can be zero or one. (Goodfellow et al., 2016) (Suzuki, 2011). We can mathematically express the functionality of an artificial neuron as follows:

$$y(k) = F\left(\sum_{i=0}^{m} w_i(k) \cdot x_i(k) + b\right)$$

Where:

- $x_i(k)$ is input value in discrete time k where i goes from 0 to m,
- $w_i(k)$ is weight value in discrete time k where i goes from 0 to m,
- b is bias,
- F is a transfer function,
- $y(k)$ is output value in discrete time k.

(Suzuki, 2011)

Not only the neural network but also the perceptron itself is inspired by the functioning of the brain. If we take a closer look at the human neuron its reception and transmission of signals works on a similar basis. The human neuron is also composed of 3 parts. Dendrites, the nucleus of the cell or soma and the axon. The dendrites receive electrical signals. The nucleus of the cell or soma processes these signals and when the boundary is crossed sufficiently, the action potential is transmitted through the axon to the final part of the axon namely the synaptic cleft. Which, with the help of neurotransmitters, transfers the action potential to another neuron. (Suzuki, 2011)

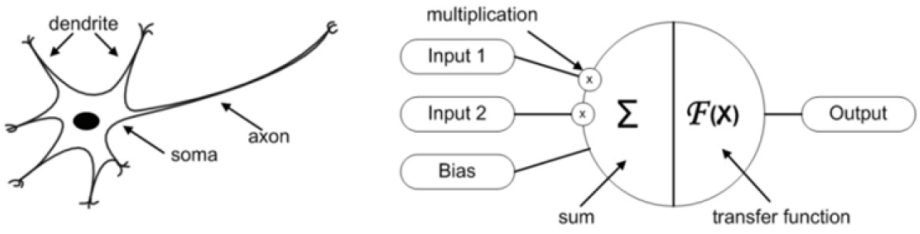

Figure 1.2: Comparision of perceptron and neuron Suzuki (2011)

However, the problem of a single neuron is that it can only solve linear mathematical problems, when we combine individual neurons into layers we gain the ability to solve problems in a non-linear dynamic way with respect to local factors. This is where we come to NLP. In NLP we are working with language, which is not just the words themselves that is transformed into a number that matters, but the order, context, language, human preferences, and other factors that are important for understanding the text (Eisenstein, 2018). Therefore is important way of work ANN with text. However, there are different types of neural networks, so we will introduce the basic types of neural networks. The basic division of neural networks can be based on the architecture of neural networks. In the field of NLP, we can mention two basic types of neural networks architecture Feed-forward network and Recurrent neural network. The difference between these two architectures is just the input processing. In FFN, the input flows in one direction, on the other side of Recurrent neural network where the input can also flow in the opposite direction. (Suzuki, 2011). Further, we can mention Gated Recurrent Unit and Long Short-Term Memory, which are improvements of recurrent neural network and the Convolutional neural network. Which are a special type of FFN.

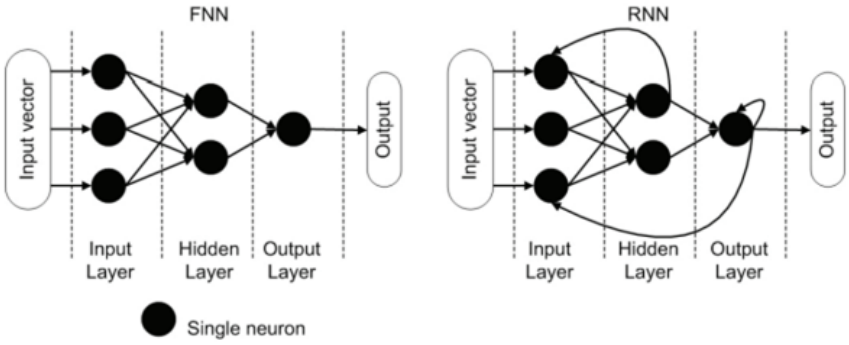

Figure 1.3: Comparision of perceptron and neuron Suzuki (2011)

Within the Transfomer architecture, we use a FFN because of nonlinearities. Nonlinearities enable the transformation of inputs into complex outputs. Where a two-layer FFN in which the first layer (or hidden layer) introduces nonlinearity via $\mathrm{ReLU}(\cdot)^2$ and the second layer involves only a linear transformation, because the goal is to transfer complex relationships between words into linear form. Which we can express using the formulas:

$$\mathbf{H}_{\text{out}} = \mathbf{H}_{\text{hidden}}\mathbf{W}_f + \mathbf{b}_f \tag{1}$$

$$\mathbf{H}_{\text{hidden}} = \mathrm{ReLU}(\mathbf{H}_{\text{in}}\mathbf{W}_h + \mathbf{b}_h) \tag{1.1}$$

where $\mathbf{H}_{\text{hidden}} \in \mathbb{R}^{m \times d_{\text{ffn}}}$ is the hidden states, and $\mathbf{W}_h \in \mathbb{R}^{d \times d_{\text{ffn}}}$, $\mathbf{b}_h \in \mathbb{R}^{d_{\text{ffn}}}$, $\mathbf{W}_f \in \mathbb{R}^{d_{\text{ffn}} \times d}$ and $\mathbf{b}_f \in \mathbb{R}^d$ are the parameters (Xiao and Zhu, 2023).

1.2.2 Attention

Transformers differ from conventional neural networks due to attention. The inherent attention mechanism allows the model to highlight important sequences during further processing Vaswani et al. (2017). The mechanism in question can be likened to human word processing, where we need keywords to form a judgment. Within the attention mechanism we must mention 2 important mechanisms that together create attention: self-attention, multi-head attention.

Self-attention

Unlike other neural networks, the Self-attentiveness mechanism is concerned with the correlations between a token and all other tokens in the token processing part, where the result is a weight sum of importance for all other tokens.

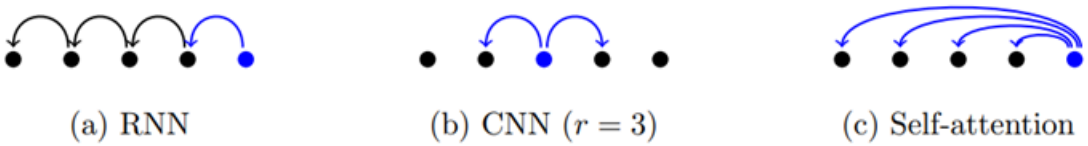

Figure 1.4: Difference between attention and other ANN (Xiao and Zhu, 2023)

.

To calculate the relationship between tokens with each other, we create 3 matrices (K-keys, Q-queries and V-values). These matrices are created using weight matrices that are adjusted during learning, thus learning which tokens are important. We then apply Scaled Dot-Product to the created matrices,

$$\text{Att}_{qkv}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right)\mathbf{V}$$

Where Query vectors Q are multiplied by the inverted Key matrix $K^\top$ to compute the dot-product score. The dot-product of keys and queries for each dimension is divided by $(\sqrt{d_k})$ to obtain the similarity between keys and queries. The entire score matrix is passed through the *softmax* function, normalizing the values to a probability distribution. The output is computed as a weighted sum of the values from the Value matrix V, with the weights coming from the softmax scores (Vaswani et al., 2017; Xiao and Zhu, 2023).

Multihead attention

The authors have shown that it is advantageous for obtaining multiple perspectives to create multiple h times with different, learned linear projections, which allows the model

to learn from multiple subspaces of lower-dimensional features. Therefore, multi-head attention was applied, where one head processes the entire set of tokens with different weight matrices to create (Q,K,V) matrices.

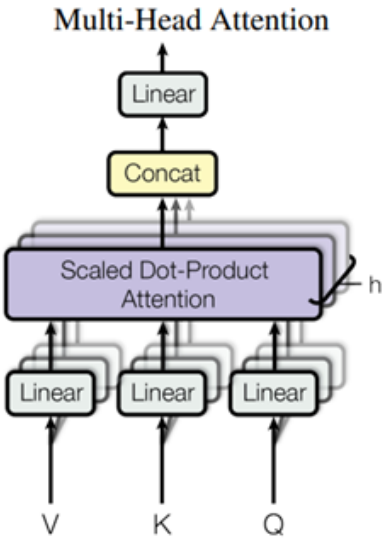

Figure 1.5: Visualization of multi-head attention (Vaswani et al., 2017)

.

1.2.3 Architecture of Bidirectional Encoder Representations from Transformers (BERT)

BERT (Devlin et al., 2019) is a specific type of Transformer architecture, and we mention this specific architecture because, will be used for a practical part of the diploma thesis. In contrast to the Transformer architecture, BERT uses a different architecture and provide new methods to acquire representations of language. Self-attention in generative pre-trained transformer (GPT) (Radford and Narasimhan, 2018) computes attention weights based on information from the left or right context of the current word in the sequence. That is, the model considers the past and future when making predictions for a given word in the sequence, which help generate coherent text. GPT has only decoder part of the transformer. On the other hand, Bidirectional self-attention in BERT: obtains a contextual representation of the word from both directions, allowing the model to capture the dependencies and relationships that lie both before and after each token. BERT has only encoder part of transformer, and different number of neuron layers and attetnion heads. Authors of BERT created two versions: BERT-Base with 12 FFN, 12 AH with total parameters 110 M and BERT-Large with 24 FFN, 16 AH and with total parameters 340M. The BERT also process input differently. The BERT-Base is composed of sequences of 512 tokens. Input text is

divded by these chunks, and at the beginning and end are classification tokens ([CLS]) while a token ([SEP]) is added to separate the sentences in the sequence. In addition to the positional embeddings, a segmentation embedding is added that specifies the position of the token in a given sequence. In the token creation framework, Bert uses the WordPiece tokenizer. Wordpiece tokenizer is a kind of sub-word tokenizer that is based on calculating a score from the frequency of a pair divided by the frequency of the first and second character separately (Schuster and Nakajima, 2012). Thanks to the Transformers library (Transformers, 2024) in python, we can work with the detailed specifications of the BERT model, and during the creation we can set the individual model parameters such as the number of attention heads or the number of neural layers via the model.config attribute.

Pre-training and Finetuning

Preparing the Transformers architecture for a specific task involved training the language model from the ground up (Vaswani et al., 2017). As part of the creation of the GPT architecture (Radford and Narasimhan, 2018), the authors created methods to train the models namely Unsupervised pre-training and Supervised fine-tuning. Unsupervised pre-training is used to gain language skills and Supervised fine-tuning is used to gain task-specific skills. Unsupervised pre-training can have different methods prior to language acquisition. Within the aforementioned BERT architecture. The pre-training techniques are Masked LM and Next Sentence Prediction. In Masked LM, 15% of the text is removed and BERT attempts to predict the token. In Next Sentence Prediction, BERT tries to predict the next sentence to understand the relationship between two sentences. Then after this pre-training, we can perform fine-tuning of the specific task. From a technical point of view, the last layer is replaced and replaced with a task-specific layer. (Radford and Narasimhan, 2018). In the specific case, we can talk about training on a small, specific dataset with labels to determine a specific emotion. The model with the new last layer modifies the weights of the pre-trained BERT model to specialize the emotion classification by optimizing the model for the specific features of the emotion dataset. With fine-tuning, the model better understands and predicts emotions in new, unseen text based on patterns learned during the fine-tuning process (Devlin et al., 2019).

1.3 Slovak language and Current LM

In the context of training LMs, we can observe a clear trend of training language models with high-data-intensity languages. According to a study by (Joshi et al., 2020), we can observe 7 languages in the forefront that contain a large amount of data in the online

space due to which they have enough data to build large language models. However, Slovak is not one of these languages. The lack of text in the online space in languages like Slovak forces language modelers to create Multilingual Language Model (MLM) to cover also Low-Resource Languages (LRL).

There are several ways of creating MLMs. According to (Qin et al., 2024) MLM creation is based on parameter-tuning alignment, which uses model parameter tuning, which can be done in several ways, namely pre-training, supervised finetuning, reinforcement learning from human feedback, and Downstream Finetuning Stage. Some ways of training involve applying different languages from the beginning of pre-training. An example of this method is From-scratch Pretraining Alignment, where LRL are inserted by equitable data sampling from each language during pre-training. The XGLM model Lin et al. (2021) used LRL to expand multilingual capabilities and outperformed GPT-3 in multilingual common sense reasoning, where equitable data samplings of more than 20 languages were added during from-scratch pretraining Equitable use of languages can lead to balanced use of multiple languages (Zhong et al., 2024) (Ozsoy, 2024). On the other hand, some methods only add to the already basic language obtained from pre-training using supervised fine-tuning (Qin et al., 2024). The positives can be seen in lower cost but on the other hand, the ability to use other languages is not the same as using the dominant language in the model (Etxaniz et al., 2023; Jin et al., 2024; Ozsoy, 2024). At the same time, LRL have a higher chance of overcoming GPT-4's safety filter by more than 70 percent compared to English (Yong et al., 2023).

On the website `https://huggingface.co/models?language=sk/` [1], we can observe 636 available models that contain the Slovak language. However, most of these models are multilingual and are part of a large language cluster. Examples of multimodal models are, for example, openai/whisper-large, which is used for automatic speech recognition. The model was trained on 680k hours of labeled audio data from different languages, and of these only 117k were from a group of 96 languages (Radford et al., 2023). Another example is the EUROLLM multilingual model, which is made up of 50% English (this proportion will decrease as the pre-training process progresses), and the remaining percentages are split between the other languages (namely 32 languages). The percentage representation of Slovak language was around 1 percent (Martins et al., 2024). The models mentioned above serve as an illustration of how Slovak language has a small representation compared to other languages. Of course we can point to the low amount of text. Within the Slovak language models we can mention SlovakBert and the Slovak-gpt-j models (162M, 405M, 1.4B. parameters). For SlovakBERT pre-training, sources from Wikipedia (326 MB of text), Open Subtitles (415 MB), OSCAR 2019 corpus (4.6 GB) were used and the largest part of the dataset

[1] Accessed: 19.09.2024.

was crawled Slovak pages with .sk domain (17.4GB) (Pikuliak et al., 2021). If we compare the parameter sizes or the amount of data for LMs such as GPT-3 and Slovak LMs such as slovak-gpt-j we can see a several-fold difference in favour of GPT-3. Therefore, we consider the development of CL methods for LRLs as the key for LM development despite the lack of data.

1.4 The use of cognition in LM

In the previous chapter, we pointed out that there were a greater number of language models in Slovak that were, for the most part, multilinguals. Insufficient resources for pre-training and fine-tuning can be challenging in terms of the financial and personal resources needed to create it, while at the same time research in the field can be severely hampered for the very reasons mentioned above. On the other hand, proposed language models, such as the aforementioned Whisper or Eurollm, can be difficult to manipulate. We can notice a certain trend where all companies such as OpenAI (gpt 3.5, gpt 4) (OpenAI, 2023) or Meta (LLama 3.1, LLama 3.2) (Meta AI, 2024) are increasing the size of their models to achieve better results. With less data with a smaller architecture, the models can be handled better and faster iteration of hyperparameters to model the language or application of methods to improve the capability of the models such as fine-tuning or soft-prompting, can take place. We can facilitate the aforementioned research or entrepreneurial endeavors by improving the ability to make the most of the available data. In comparison, Large Language Models (LLM) need significantly larger amounts of data to understand the language and subsequently fine-tune for a specific task (Warstadt and Bowman, 2022).

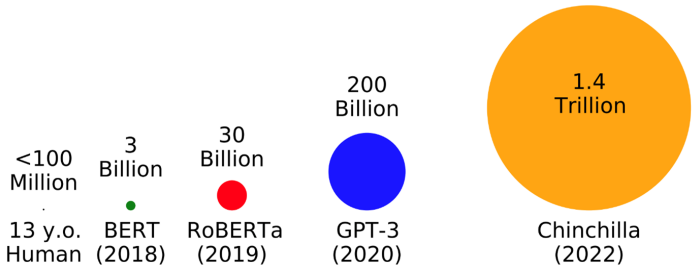

Figure 1.6: A visual comparison of the number of words processed by a 13-year-old human (<100 million) in comparison with several language models —BERT (3 billion), RoBERTa (30 billion), GPT-3 (200 billion), and Chinchilla (1.4 trillion)(Warstadt and Bowman, 2022).

As part of the BabyLM challenge, researchers are trying to improve pre-training on the amount of language needed for comprehension by a 13-year-old child. Not only the positivity in the amount of language is a clue to this research but also human ability

to work with language is still better in some areas compared to established LMs e.g. linguistic generalization abilities (Beinborn and Hollenstein, 2023). The criticism of the LLM stems from these shortcomings. And it is human characteristics such as efficiency and robustness in language processing that can be perceived through sciences such as linguistics, psycholinguistics, cognitive plausibility or cognitive modelling, where we try to adapt LM language processing to the human (Keller, 2010). Cognitive plausibility LM focus on the application of psycholinguistic variables such as reading times, gaze durations, and brain signals without much concern for developmental plausibility. Plus the formal and functional linguistic competence of LLMs misses that of humans. Which we need not see as a negative because formal linguistic competence - knowing the rules and statistical regularities of language - and functional linguistic competence - the ability to use language in real-world situations. When LMs are focused on formal linguistic competence, which only comes from language, and people are more focused on functional linguistic competence, which comes from the real world (Mahowald et al., 2024).

Which can support a given artificial LM at the three levels of the cognitive model according to Marr (2010). Marr describes the three levels of the cognitive model as computational, algorithmic and implementation. The computational level deals with the problem of what the model needs as input and what it needs as output and the constraints that may hinder the process. The algorithmic level focuses more deeply on the computational level problems by specifying the procedure and representation of a given model, and the implementation level deals with the realization of the model in the physical world or model architecture (Marr, 2010). By transforming the positive properties of human language work into LM, we aim to improve all three levels of a given model. The application will enable an architecture with fewer parameters (implementation) allow data to be processed faster (algorithmic) and create models that are closer to humans (computational).

In the BabyLM challenge we can observe several approaches to the solution. Of course some solutions were not cognitively inspired. Such as models dealing with hypertuning of parameters. We can see the positive in the application of lower parameters and less text in ToddlerBert (Cagatan, 2023), where it was possible to train more than 180 BabyBERT models with different parameters. 5M words of American-English child-directed input were used to pre-train BabyBERTa. The best trained ToddlerBERT model was used for comparison with T5, Roberta(base) or OPT-125m models based on BLiMP, SuperGlue or MSGS benchmarks. In none of the benchmarks was the model produced last and even in the BLIMP task it achieved the best score. Regarding more cognitive solutions we can mention a few. One example is the McGill BabyLM. To pre-train the given model, several methods were applied to format the text. One example is the context of the embedded text. Often times, due to the size

of the batch, the end sentences can be split into parts and their context can be lost (Cheng et al., 2023). A similarity can be found in the effect of context on children's vocabulary learning (Nagy et al., 1985). According to the authors, BabyLM challenges performance gains in (Super)GLUE but unperformed the models without such learning on MSGS (Warstadt et al., 2023b). As a final example, Cogmemlm was inspired by human remembering/forgetting, where an individual token has an activation number. The activation number varies depending on the occurrence of the token in the text. At the same time, the size of the lexicon replicates the size of the child's vocabulary. (Thoma, 2023) The evaluation of the given technique is seen positively however the results from the baselines could be due to the segmentation system or other hyperparameter adjustments Warstadt et al. (2023b). Global CL methods were used in 40% of the studies involved in the BabyLM challenge, the remaining work focused on preprocessing, hyperparameter tuning and model scaling or the use of auxiliary model and other methods. It is from these solutions that we can draw inspiration for the creation of SlovakbabyLM.

1.4.1 Learning of language

Various aspects of human language learning overlap with the learning of LM, but some are again missing. Moreover, what language-related knowledge is acquired during the learning process? In pre-training and fine-tuning, most authors do not consider human learning (Warstadt et al., 2023a). Gradual acquisition is necessary for humans to acquire adequate skills to achieve environmentally appropriate goals. We can look at this problem from several theories and perspectives. From the perspective of behavioral psychology, we can talk about the formation of behavior that is subsequently rewarded or punished (Skinner, 1958). One of the basic types of learning involves imitating other people. Within language, children often imitate the speech and gestures of their parents or peers while acquiring communication skills (Bandura, 1965). Children choose the sources of learning it is mainly parents, siblings and peers who influence our language learning through reinforcement and punishment (Poulin-Dubois and Brosseau-Liard, 2016). Imitation comes from the external environment from which input flows. It is these inputs that model the learned language and their first words are imitations of the words they hear from their parents (Tomasello, 1992), which can be compared to the token production by a BPE tokenizer, which selects and produces tokens based on a high frequency and the first tokens are not perfectly accurate (Sennrich, 2015). On the other hand, we can mention the view based on Chromsky's theory of Universal Grammar (UG), Chomsky believes that human language is governed by a set of rules or principles that are innate to the human mind. According to Chomsky, this innate linguistic ability allows humans to produce an infinite number of grammatically correct

sentences using a finite set of rules (Chomsky, 1980, 2014). These two theories outline for us one of the fundamental problems in language learning, namely the existence of innate and acquired language abilities. However, LM does not have the possibility of acquiring something innate and learns by experience. Therefore, in this section, we will deal with the acquisition of the ability to use language through learning.

If we go deeper into the child's development, we can speak of a basic understanding between two subjects, namely the child and the parent, which stems from gestures or sounds that become intentional communication signals, where the child moves from the role of a passive participant to that of an active user of the means of communication, and they anticipate each other's behavior based on these gestures. Research on monkeys and chimpanzees has shown that these means of communication differ from individual to individual. A practical example is in chimpanzee play, where one juvenile chimpanzee makes a specific gesture (raising its hand and striking) and the other initially does not understand the gesture but after some repetition learns to anticipate the attack (Tomasello, 1992; Halina et al., 2013). This mode of learning has also been demonstrated in early childhood (Marentette and Nicoladis, 2012) and we can see the application of this learning in the learning of communication of humanoid robots (Spranger and Steels, 2014). From neuroscientific perspectives, we can point to the critical period hypothesis. The learning of the primary language at given critical periods is crucial to language acquisition, where it is the separation from parents at an early age that may reduce the ability to communicate (Siahaan, 2022).

The foundations of language acquisition in LM can be identified with statistical learning theory focusing on human language acquisition. Statistical learning theory points to the human ability to unconsciously perceive patterns and regularities in the environment, and to learn and acquire new information because of this ability. In the context of language acquisition statistically significant occurrence of specific inputs, specifically letters or syllables occur more frequently side by side and children perceive this (Sherman et al., 2020). This sensitivity can be seen as early as 8-month-olds, when symbol frequency helped children segment fluent speech (Aslin et al., 1998). Different frequencies of letters and syllables are also found in the Slovak language (Štefánik et al., 1999). This term can be also found in computer science (Vapnik, 1999). But we will focus on linguistic perspective. At the same time, Statistical learning also helps to identify different categories or groups of objects or stimuli based solely on exposure to examples from these categories (Maye et al., 2002). However, it is important to mention that individuals can be sensitive to certain types of statistical patterns. According to Thiessen et al. (2013), we can draw on three types of statistical sensitivities: conditional probability, distributional probability, and cue-based statistics.

Conditional probability is the probability, that event Y will happen given the information that another event X has occurred. In human processing, we can notice various difficulties with higher levels of text. Conditional probability is represented in language processing by predictions of the following words. However, surprise theory claims that the occurrence of a non-predicate word increases the processing difficulty and hence the processing time. This processing time can be seen in the higher reading time of an individual (Futrell and Levy, 2017). Thus, from a conditional probability perspective, unexpected words or characters in a text have a low probability of occurrence and thus disrupt reading fluency. Next probability is distributional. Distributional statistics capture the central tendency and prototypical characteristics of a set of elements. This implies that the number of occurrences of specific linguistic features influences the comprehension of those linguistic features in the future. Self-paced reading of non-linguistic and linguistic stimuli targeting patterns of verb sequence in German by L2 learners demonstrated a temporal difference in reading non-linguistic linguistic stimuli (Perruchet and Poulin-Charronnat, 2012). Which may indicate the effect of prior experience with a specific grammatical form on shaping our cognition. From a statistical language learning perspective, we may equally predict individual differences. The differences stem from variability in vocabulary gain, early environmental influence or different developmental trajectories (Kidd et al., 2018). From a phonetic perspective, we can observe an increased ability to discriminate /d/ and (unaspirated) /t/ in infants when these letters were exposed to a bimodal distribution of sounds. Which means that clear exposure to the characters may lead to better recognition of these characters which underlines the importance of statistical sensitivity (Maye et al., 2002). The last statistical sensitivity mentioned is the sensitivity to the occurrence of a perceptual feature indicating another occurrence of a feature that is not accessible to us but we assume is there. Cue-based statistical learning leads to the evaluation of individual cues and the search for those cues. Here we can mention making associations , which have an impact on our memory or word recall (McNamara, 1992). We can also find this statistical principle in the Bert language model. When the probability of the word Airplane in the unfinished sentence "I want to be _____" increased if the word Airplane was ingested in the previous conversation (Misra et al., 2020). This statistical sensitivity does not affect the formation of word associations but also the perception of various external characteristics of objects or the understanding of human emotions (Rakison et al., 2008).

1.5 Cross-linguistics differences between English and Slovak

One of the main reasons for doing this work is the difference between the Slovak and English languages. As it has been already mentioned, the application of CL does not exist in other languages and it is the application of CL and other active learning methods to LRL that can be the key to the development of the field of NLP. However, the Slovak language is different in many aspects. The complexity of languages can be inferred from a number of basic components of a language, but since we focus on text, we will mention the attributes that make up the written form of a language and which make languages significantly different from each other (Morphology, Syntax, Semantics). and finally, we mention ontogenetic language development and interlanguage differences.

1.5.1 Morphology

In linguistic terms, morphology refers to the study of the internal structure of words, which includes word formation or inflection. Comparing English and Slovak in morphology, we can show a significant difference in Syntheticity in favour of Slovak (Horsch, 2021). Slovak has a higher number of consonants, a higher average number of morphemes in inflection and is inflection-based. An example of more complex inflection is the use of inflectional endings. The accusative is used to form the subject, whereas in English the focus is on derivational morphology, where we use specific suffixes to form the word type (-ed, -ing). It is the changing word bases and modification of words that are used to express different grammatical categories (Panocová, 2021). A study of machine translation from Slovak to English points precisely to the ambiguity of the use of morphemes in inflection. In some cases the inflectional morpheme of a noun may indicate its gender, but in many cases of case suffixes this is not the case. The authors give the example of the morpheme '-a', where it is used for the masculine 'ten hrdina' (the hero), the feminine 'tá žen-a' (the woman) and also for the neuter 'to dievč-a', (the girl) (Welnitzová and Munková, 2021). The significant frequency of suffixes compared to English may increase the number of tokens in the Slovak language model.

1.5.2 Syntax

Syntax focuses on the relationships of sentence constructions found within a sentence, including the relationships between sentences within clauses (both simple and compound). When comparing languages again, Slovak syntax can be considered more complex. From the derivational morphology in English we can conclude that English

needs more words to form grammatical categories but on the other hand they are based on a fixed word order. A sentence in Slovak (Adam maľuje stenu- Adam paints wall) can be expressed by switching the object and subject but in English the word order changes but also words are added (Stenu maľuje Adam- Wall is painted by Adam). (Dolník, 2010). Therefore, the authors Welnitzová and Munková (2021) consider the Slovak-English translation as more difficult with a higher probability of errors. In addition to the looser syntax, they also mention the unexpressed subject or the lower number of tenses compared to English.

1.5.3 Semantics

Semantics focuses on the expression of state or action in a sentence. A study by Urbániková (2010) compared Slovak and English words in the Swedesh list and outside it (Swedesh list is a list of provisionally universal terms). From this list, there is an overlap in at least one of the meanings, sometimes in most or even all meanings; on the other hand this overlap is generally lost. The authors have given an example of the meaning of the metaphor of the word "lambs", which resemble white fluffy puffs, literally "lambs", in their colour and consistency of waves. However, this meaning is not found in English. On the other hand, we can mention the English example: " raining cats and dogs", which, on the other hand, is not found in Slovak. From specific examples, we can move on to statistical data. Of the 346 words examined, 161 have approximately the same number of meanings. There were a total of 66 Slovak words and 97 English words that had more meanings than their English counterparts. Again, the authors attribute these results to the analytical nature and to the plurality of English. Meaning, the English language uses more of the same words in different contexts. Of course we can see further differences, namely in the use of the verb or noun as the centre of expression of a state or action. English makes more use of the noun. On the contrary, Slovak relies on verb inflection and the reason for this is the English finite verb, which does not have extensive inflection in contrast to Slovak (Bérešová, 2016). An example is the verb: "He decided" in English, however, we have to say "He decided" to indicate the gender of the person who decided. For the problem of statistical learning, this reduces the frequency of the same words in other contextual meanings, and at the same time the inflected words carry more meaning.

1.5.4 Ontogenetic language development and interlanguage differences

The importance of creating the BabyLM dataset lies mainly in the communication between mother and child. Our work will attempt to establish this conversation through LLM hence it is important to highlight the differences and commonalities in the emergence of speech across language differences. In a study of children's vocabulary in seven languages, Maital et al. (Spanish, Dutch, French, Hebrew, Italian, Korean and American English), vocabulary similarities were demonstrated. A larger part of the vocabulary was made up of nouns. Waxman et al. (2013) cite object mapping and the identification of object categories as the reason for the higher number of nouns in the child's vocabulary. The acquisition of verbs into vocabulary is more context and language itself based due to the higher abstractness of base verbs compared to nouns. This requires increased parental interaction and responsiveness to the child's reactions (Snow, 1977). Further, frequent repetition of words (Newman et al., 2016), child-parent interaction during play or fictive play (Tamis-LeMonda and Bornstein, 1994), and a gradual increase in the diversity of grammatical forms, more abstract concepts with increasing age are important (Snow, 1977).

From a developmental point of view, the above facts can be considered identical, but children's speech errors in language acquisition are different across language groups. An example would be the ability to form the passive voice. A study on 11 languages showed differences not only in the task of forming the passive tense but also in the production of errors in the use of the passive tense, where German and Dutch differed from the other languages (Armon-Lotem et al., 2016). Cross-linguistic differences also emerged in specific language impairment, where children with the same age had a different vocabulary of verbs (Leonard et al., 2004). However, if we look at this from the perspective of Slovak language. The emergence of the fall in Slovak children, when in infant babbling children start to use fall suffixes despite not using the full word (children's age was up to three years old) (Kesselová, 2014). This complexity of inflection can also be observed in the acquisition of words, where English-speaking children produced a higher amount of verbs compared to Slovak children but the difference in verb comprehension, nouns and noun formation was lower in percentage terms (Haman et al., 2017).

1.6 Curricullum learning

The improvement of learning can be based on a certain arrangement and design of curricula that contain elements important for the acquisition of a certain skill that shape our thinking (Skinner, 1958). The gradual shaping of a particular ability (in this case language) would not necessarily be part of the human species but we can apply them to language models (Bengio et al., 2009).

Bengio compiled three conditions that the curriculum must adhere to (Bengio et al., 2009; Wang et al., 2021) :

1. The first condition: "Curriculum Learning (CL) is a sequence of training criteria over T training steps:
$$C = \langle Q_1, \ldots, Q_t, \ldots, Q_T \rangle$$
Each criterion Q_t is a weighting of the target training distribution $P(z)$." (Bengio et al., 2009)

   In practice, this means that the CL procedure is supposed to increase the amount of more useful information with increasing learning length.

2. The second condition: The weight for any example increases, i.e.,
$$W_t(z) \leq W_{t+1}(z) \quad \forall z \in D$$
   Meaning, the amount of input is to increase in quantity, thus increasing the weight of the specific feature to compute.

3. The third constraint talks about the final stage of training, where we train on a uniform weight of dataset:
$$W_T(z) = 1 \quad \forall z \in D$$

Of course, the individual conditions are individually fulfilled according to the task in which the CL is used. The second and third conditions may not be fully satisfied in reinforcement learning, due to the agent learning within the execution of a single action (Wang 2021). This procedure can be likened to various optimization algorithms such as continuation methods (Allgower and Georg, 2012), simulated annealing (Kirkpatrick et al., 1983), or Genetic algorithms (Holland, 1975). These algorithms are used to solve optimization problems by exploring the search space and finding optimal or near-optimal solutions that have multiple optimization minima. In the context of transfer learning, an optimization problem can be viewed as a cyclic generalization of previously learned language and knowledge that is acquired through training on a simpler text. Text consisting of more complex words in terms of morphology or text containing noise

may be less easy to process and thus will take longer to process (Bengio et al., 2009). From an ontogenetic point of view, CL can be likened to the developmental stages of Piaget (2000). The theories in question speak of cognitive development as a gradual progression, whereby at different ages, which differ in the processing of stimuli, which differ in the size, difficulty and type of stimuli. A simple example is the difference between a toddler and a child at age 15. While a toddler only reflexively responds to basic stimuli. A child at 15 not only understands but can express himself using abstract concepts. This gap is just filled by the aforementioned stages of cognitive development. Which we can just liken to CL. Bengio point to the evidence gathered in their work and offer two positives that arise from the curriculum. The first is faster learning. The learning agent is offered only a foundation to learn a given skill, resulting in less redundant text. Another is guided learning, where we can guide the agent to better generalization during deeper learning control (Bengio et al., 2009). On the other hand, we can point to the opposite extreme, namely the lack of data. An example is the exclusive-OR function, based on which we can further conclude that learning with little data may not be sufficient (Elman, 1993).

1.6.1 Types of Curriculum learning methods

For better understanding and use of the method, it is necessary to define what data is "better" and how to select these data, which is related to different algorithms and selection methods that change parameters or select specific data during training. In one of the meta-analyses regarding the topic we can see a division into up to 7 different types of CL however Self-Paced learning (SPL) is combined into Self-Paced Curriculum learning (SPCL) therefore we will not mention SPL but only SPCL therefore this subchapter is made up of 6 different types of CL which are used in machine learning according to Soviany et al. (2022).

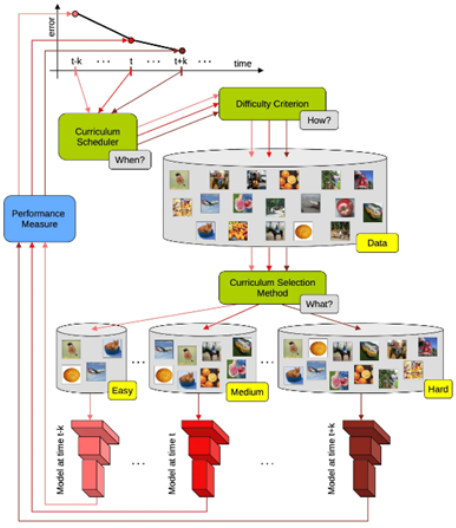

Figure 1.7: General framework for data-level curricullum learning (Soviany et al., 2022)

One of the most basic CL techniques is Vanilla CL, which we mentioned in the introduction. The given learning method has been used to process geometric figures, where we first process lines and only then the figures composed of lines. Furthermore, this technique was used to train LM, where they split the wikipedia dataset into parts and accepted only the parts that contained the 5000 most frequent words. If a given part of the dataset did not contain a given word, it was not used for the current training. Thus, the training set became larger after each pass through Wikipedia (Bengio et al., 2009). The aforementioned active learning method triggered a revolution in the field and other CL techniques developed vanilla CL. The selection and layout of text can be based on different ways. The combination of specific text features as rules for judging text ordering can be seen in Bayesian optimization for task-specific word representation learning (Tsvetkov et al., 2016). The authors created three important factors: diversity, simplicity, prototypicality. Diversity measures the distribution of different data types and they used, for example, Number of word types or Type-token ratio. Simplicity measures the complexity of paragraphs in terms of phonology, lexical

and syntactic complexity. For example, Verb-token ratio or Noun-token ratio were used. Prototypicality captures semantic elements that come from cognitive linguistics and children's acquisition of language, such as word abstractness.

$$s(X) = \mathbf{w}^\top \boldsymbol{\phi}(X) \tag{1.2}$$

Where $\boldsymbol{\phi}(X)$ represents the linguistic properties of the paragraphs and $\mathbf{w}$ are the learned weights for these features.
where:

- $\boldsymbol{\phi}(X) \in \mathbb{R}^d$ is the vector of linguistic features for part X, drawn from the criteria.
- $\mathbf{w} \in \mathbb{R}^d$ is the weight vector *learned* (via Bayesian optimization) to balance these features for the target task.
- The given text ordering was found to be more efficient compared to the plain text ordering.

A more advanced technique is SPCL, where we work with the learner (Jiang et al., 2015). In other words, a given order of data dynamically adapts to the pace of the learner which solves the regularization problem by applying SPL, which adds a regularization value based on a certain methodology. In Vanilla CL, the learning scheme is pre-determined by prior knowledge and subsequently remains invariant. If we focus on brain functioning and take the free energy principle. The organism tries to minimize the entropy of its sensory states. The overall functioning of the organism focuses on this process. We can see this in the enforcement of sampling of sensory data that is consistent with the current representation Friston (2009). From the above free energy theory, we can assume that the selection of the next option is based on the previous embedded representations, which is what the combination of these two methods secures and thus more closely replicates human language processing. Mathematically, we can express this model as follows:

$$\min_{\mathbf{w},\mathbf{v}\in[0,1]^n} \mathbb{E}(\mathbf{w}, \mathbf{v}, \lambda, \Psi) = \sum_{i=1}^{n} v_i L(y_i, g(\mathbf{x}_i, \mathbf{w})) + f(\mathbf{v}; \lambda) \tag{1.3}$$

In SPL, the goal is to jointly learn the model parameter $\mathbf{w}$ and the sample weight variable $\mathbf{v} = [v_1, \cdots, v_n]$. $L(y_i, g(\mathbf{x}_i, \mathbf{w}))$ denote the loss function which calculates the cost between the ground truth label y_i and the estimated label $g(\mathbf{x}_i, \mathbf{w})$. Regularization factor $f(\mathbf{v}; \lambda)$ control learning pace. Where size of λ decide importance of size of lost important for model.

Balanced Curriculum Learning (BCL) in terms of our problem complements Vanilla CL for the diversity of data in our world. BCL takes into account the distribution of the data obtained from our world, where a few majority categories occupy the majority of

the data, while minority categories contain a limited number of samples Soviany et al. (2022). In general, however, we can talk about ordering datasets based on a balanced distribution, where we order the data from the simple to complex but also consider the distribution of the data (Wang et al., 2021). In a study Zhu et al. (2022), they focused on supervised contrastive learning, which is used to find co-salient regions of images that are relevant to a specific task (detection of foreground objects). But the authors warned on long-tail data (images that are not completely frequent), this metric may achieve unsatisfactory performance due to the fact that the training will only focus on the statistically most frequent data. And not only within individual data but also the distribution of groups of data. Therefore, the authors combined supervised contrastive learning and BCL. The authors proposed a combination of losses that are computed within a single batch. For a single batch of data embedded to train the model, the occurrence of a given group of images (dogs, buildings, etc.) and the complexity of predicting the saliency regions of the images were taken into account (Zhu et al., 2022).

Teacher-student CL divides the task into two parts: the first part consists of the student's performance of the given task and the second part consists of an auxiliary function, in other words, the teacher changing the student's behavior based on the particular function or the reward received. In practice, this may mean using the secondary model to train the primal model, which in training, we use to find the part of the text that produces the lowest value of the Lost function. The retrieved portion of text with the lowest possible loss function value is applied to the primary model Soviany et al. (2022). An example is the use of unlabelled data with labelled data. The procedure is again divided into two overlapping parts. As the training time increases, the amount of unlabelled data increases, which increases the training difficulty Zheng et al. (2019).

Implicit CL is created using side-effect methodologies and we do not create a fixed ordering of easy and hard methods. An example is the convolution of the output activation maps of each convolutional layer with a Gaussian kernel. During training, the variance of the Gaussian kernel is gradually reduced, allowing more and more high-frequency data to pass through the network. Which translates to learning based on the frequency of given features Sinha et al. (2020) An example of the use of the method in NLP is the use of cosine similarity Han and Myaeng (2017). This assumption is related to the ease of learning facts that are partially known as purely new facts. The authors created a Tree-Structure and used Ward's method Ward Jr (1963) as a data extraction algorithm which they subsequently applied to RNN in sentiment analysis task, Ward's method minimize the total within-cluster sum of the squared errors and the hierarchy is constructed by backward observation of K-1 cluster formation with K nodes . The training procedure is based on the gradual introduction of subsets until the last stage is reached. The result was an improvement of 3 percent in accuracy on the given tasks over sequential CL.

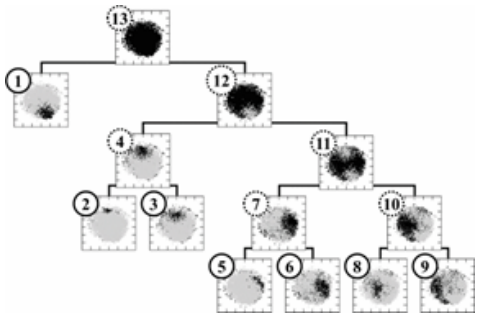

Figure 1.8: Description of selection data for semantic similarity (Soviany et al., 2022)

Progressive Curriculum Learning (PCL) which is not related to individual data sets but to the modification of model parameters (Wang et al., 2021), such as changing the Dropout function for neural networks. Dropout increases with increasing training time, thus increasing the abstractness of a given data representation (Morerio et al., 2017). Applications can also be found in reinforcement learning, where the agent's problem of finding the fastest way out of a maze (Sokoban problem) was solved using a multi-hand bandit that assigned values to each state and a CL method that selected an option at the agent's solvability boundary, thus providing the RL agent with a significant learning signal (Feng et al., 2020). The application of PCL in NLP can be seen in the modification of the LTG-BERT model, where the authors allowed for each transformer layer to select which outputs of previous layers to process, however after modifying this model the transmission changed where ultimately the n-1 layers had the highest impact. This solution won the Babylm challenge 2023 in the Strict track (100M words) and Strict-Small track (under 10M words) categories (Charpentier and Samuel, 2023).

Chapter 2

Methods

The main objectives of the thesis are data acquisition and testing of the developed CL methods and application of this solution to LM with the same number of data as needed for language acquisition of a 13 year old child. To achieve this goal we used several methods for data mining such as web-crawling, web-scraping, text generation, or in case of unsuitable conditions also manual data collection, so in the following subsections we will mention the data collection methods, what methods and tools we used, how we used these tools to mine specific data and how we cleaned and processed the data so that it is usable for the creation of the LM. Since we want to reproduce the Babylm challenge (Warstadt et al., 2023a), we created 6 subdatasets, which we then sized for our testing needs and combined into the final dataset. We applied the created small-strict dataset (10 million words) to test the created CL solutions, by using the sorted dataset to pre-train the BERT architecture. We then subjected the given model to tests to compare the quality of the generated solutions. Subsequently, we applied the positive CL ranking methods to pre-train the BERT model. To test the application of each CL method on a larger scale.

2.1 Data collection

In this subsection, we will look at the technical method of data extraction, because of the lack of data and the specific requirements of each sub-dataset we decided to use the following methods: web-crawling, web-scraping and text generation, but if none of the methods did not yield specific data we used an existing dataset. For data mining we used the Python programming language (version 3.11.6) and the following libraries: scrapy (version: 2.11.1), beautifulsoup (version: 4.12.3), requests (version: 2.32.3), openai (version: 1.35.10) The above mentioned libraries in the python programming language are freely available.

2.1.1 Used Methods

Web-crawling

The word "crawler" in this context can be described as the ability of a created program (otherwise known as a spider or bot) to dynamically and independently crawl web pages without a specific and final goal, and then explore and exploit the capabilities of individual web pages. At the beginning, one or more websites must be predefined. The role of the program is to crawl these websites and search for other websites. It uses the URL of the web page for this task. If the spider finds a new page, the algorithm redirects the spider to the found page, where the process is repeated again, creating a network effect when discovering new content. The programmer's role is to implement various controls that restrict, direct, or command how the program should behave while also influencing the configuration of the spider, such as by setting the frequency and speed at which it collects data so that it does not overwhelm the servers with too many requests. The primary goal of this method is to discover sites that we do not have oversight of or do not have a sufficient number of fixed sites where we could use web scraping (Vanden Broucke and Baesens, 2018).

Web-scraping

Web-scraping focuses on the deeper processing of content on a website. The process focuses on crawling the web page and extracting data that is publicly available. We can verify access to that content in a robots.txt file, which defines what parts of the page are allowed to be accessed by automated tools. At the beginning of the web-scraping process, it is necessary to specify the web page from which the data will be extracted. Web scraping uses the HTML of a given page and the programmer's job is to analyze and determine in what HTML elements the necessary content is located and set up a given program to extract that content using the HTML that . The role of the program is in turn to extract that content and return it in a format (json, txt) specified by the programmer (Thomas and Mathur, 2019).

Text generation

Part of NLP is Mechanical text generation, which uses advanced tools (such as neural networks or transformers mentioned in the literature) to generate text but especially LLMs. Based on the context obtained during pre-training and fine-tuning and subsequent prompts, LLMs are able to generate text based on the input data and context. Mechanical text generation includes various tasks , such as open-ended text generation, summarization, translation, paraphrasing, and question answering. The role of the programmer is the design of prompt that is related to the generation of specific

data (Becker et al., 2024)

Manual selection

In critical cases, we decided to select sources from websites using manual selection. The manual selection of data was done under certain predefined conditions and only then the data collection was done. We used this method if the use of the previous data extraction methods would have resulted in lower quality of the extracted data and at the same time the size of the possible sources (subpages) was small enough to be manually searched. Situations where the manual selection method was used were a high incidence of grammatical errors or the inability to retrieve data through previous methods.

2.1.2 Used tools

Scrapy

A scrappy library has been selected for web-crawling and web-scraping. The comprehensiveness of the given library allows us to use it easily and quickly when crawling the web. When working with the given library, we have used several elements of the given library. For web-scraping we used Selectors, which retrieve data using XPath. XPath is a language used to manipulate XML documents. to select nodes in XML documents. which can also be used in HTML for its generalization.

Next, we used a command: scrapy shell <URL>, the command used through the console and allows you to browse a given web page using selectors. For web-crawling, we created custom classes that inherited the properties of class CrawlSpider and LinkExtractor.Class CrawlSpider allows to automatically follow links on a page according to defined rules. Class LinkExtractor is used to cull links from web pages based on defined rules. By applying a regex library or by directly specifying web pages, we define which links should and should not be followed. In addition, Class LinkExtractor has a callback, thanks to which we can apply specific requirements from web-scraping.The library has been selected for web-crawling and web-scraping. The comprehensiveness of the library allows us to use it easily and quickly for web crawling. While working with the given library, we used several elements of the given library. For web-scraping we used Selectors, which retrieve data using XPath. XPath is a language used to manipulate XML documents. to select nodes in XML documents. which can also be used in HTML for its generalization.

The overall skeleton of the given functionality of the Scrapy library consists of creating a project that creates the following files: items.py, middlewares.py, pipelines.py, scrapy.cfg and a folder of spiders. In the given files we can change the web-crawling

behavior or the data processing to the final form. However, we only modified the settings.py file, where we set the time when the crawler should send the request to the next page or set the number of requests per domain as needed.

Beautifulsoup4

The Beautifulsoup4 library, used for parsing HTML and XML documents in our case, was used for web-scraping. Beautifulsoup4 loads an HTML or XML document into a tree structure that allows easy browsing and manipulation of the content. The programmer works with Beautifulsoup4 by identifying the HTML elements containing the necessary data. In conjunction with this library, we also used the requests library, which helps Beutifoulsoup4 retrieve HTML pages by allowing HTTP requests to be sent and responses to be handled.

Openai

The openai library was created by OpenAI company. The company has created several custom multi-language LLMs, and the library openai allows working with created models, which can be used for text generation, not only in written but also in audio form (Whisper model OpenAI (2022)). GPT-4o is the model that was used for the creation of our sub-datasets. The model uses API calls to LLM and is cost-effective (the cost at the time of generation was $3/1 million input tokens and $15/1 million output tokens and a total of 8192 ouptut tokens per output (OpenAI, 2023).

2.1.3 Process of collection data

The resulting dataset contains 6 sub-datasets. While each Sub-dataset had a different way of using each tool, the following procedure was used for most of the sub-datasets. We found a suitable source with enough data for our needs and created a spider for web-crawling using the scrapy library, or created just a scraper using the BeautifulSoup4 library. In some cases we did not find a single source with a large enough amount of data. We used the google search engine to mine individual web pages containing specific data. An example is the sub-dataset of fairy tales, because they were not collected in one place, we used the google search engine and entered the search term: 'Rozprávky v Slovenčine'. We then applied our methods to the web page according to the specificity of each web page. For web-crawling, we created a separate spider for each web-page and set parameters either for the files in the Scrapy library or parameters to limit and guide our spider for a single web-page however in some cases, we skipped web-crawling using the Scrapy library altogether and only applied web-scraping using BeautifulSoup4 because there was a low number of subpages to process, or we moved

to use a for loop instead of web-crawling for better control.

2.2 Pre-processing of sub-datasets

To create the dataset, we were inspired by the Babylm challenge (2023), from where we drew the percentages of each subdataset in our created dataset. From the sub-dataset creators and the creators of Slovakbert, we took the methods of preprocessing of sub-datasets and what needs to be removed. The main working tool for Pre-processing was the regex library (version 2023.10.3) in python. The regex library is an extension of the re library (Barnett, 2024). It allows the use of regular expressions to help locate a specific substring according to the conditions conditioned by paterns. If the scraped data and processed data were not large enough to create a sub-dataset we artificially padded an already existing sub-dataset. For each sub-dataset we followed a different procedure (scraping,preprocessing) but for all preprocessing we applied Pikuliak et al. (2021):

- URL and email addresses were replaced with special tokens
- Elongated punctuation, newline character and whitespaces was reduced i.e. if there were sequences of the same punctuation mark, these were reduced to one mark (e.g. – to -).
- Whole text, if text contain signs of wrong format
- Text in a different language (Python library: langdetect (version:1.0.9))

Because of the CL metrics created and after applying the aforementioned conditions, we had to apply additional conditions that removed spaces between punctuation marks without text (such as: '. .'), or removed specific words for specific sub-datasets (such as: 'AUTOR:').

2.2.1 Pre-processing process

In the pre-processing, we used the functions that were applied based on the text specifications, at the same time the given procedure allowed us to create an order of which text is removed first and last. As part of the preprocessing, we opened the resource file edited to a single json file, where each dictionary contained two keys 'page':preprocessed text and 'url': resource name. Using a for loop, we traversed each resource and applied each function. We then saved the resources in json format.

2.2.2 Sub-dataset of Wikipedia articles

In creating the sub-dataset of articles, we used wikipedia as the main source of data, the web-crawling method, and the scrapy library. wikipedia contains additional links to the term that appears in the text. Thanks to this structure, we were able to focus on specific topics and sub-pages in Wikipedia, and thus we obtained sub-pages describing different subjects taught in primary schools in order to obtain text from different areas of life and topics that a child may encounter during the teaching process. Specifically, we used the 9 Wikipedia subpages in Slovak as homepages for the scrapy library:(`https://sk.wikipedia.org/wiki/ObÄDianska_nÃąuka`, `https://sk.wikipedia.org/wiki/Dejiny`, `https://sk.wikipedia.org/wiki/Sloven%C4%8Dina`, `https://sk.wikipedia.org/wiki/Fyzika`, `https://sk.wikipedia.org/wiki/Informatika`, `https://sk.wikipedia.org/wiki/Ch%C3%A9mia`, `https://sk.wikipedia.org/wiki/Biol%C3%B3gia`, `https://sk.wikipedia.org/wiki/Hudba`, `https://sk.wikipedia.org/wiki/%C5%A0port`)

A given scrapy spider was run 10 times, for each site separately and once with all web-sites at once, and then we removed copies of the webpages from all files. The scrapy spider had limited webpages with names: ("Zoznam:", "Kategória:","Kategórie:", "Diskusia:") to remove subpages containing different lists, due to incoherent text, we also excluded the subpage "Hlavná_stránka" so that the spider did not go to the main wikipedia webpage, and thus did not scrape the same content several times. In addition to the main preprocessing, we made further changes to the dataset. Due to the large amount of text obtained, we removed texts that were less than 55 characters to reduce the amount of sources that can be incorrectly scraped. We also used the delete_whole_line function to remove words located at the end of the text describing the sources that the page drew from, such as 'Zdroj:' or 'FILIT'.

In addition to the main preprocessing, we made further changes to the sub-dataset. Webpages that contained fewer than 55 characters were removed to remove possibly badly scraped sources that contained only a few words. The resulting sub-dataset contained 58 008 943 words and 150 463 webpages of content. The downside of this method of data mining was inconsistency, and the most recent datasets suffered the most when removing copies, suggesting that individual spiders scraped an identical set of sub-pages. Since we obtained a much higher amount of data than expected we publish data from 10 separate runs. These 10 runs contain 55 628 webpages and 22 280 194 words.

Webpages	Number scraped webpages	Number of words
History	16 616	6 483 417
Music	12 570	4 835 216
Chemistry	8 552	3 219 831
Sport	6 588	2 763 223
Slovak language	5 281	2 170 375
Biology	3 414	1 396 678
Physics	1 741	723 209
Civics	866	688 245
Full Subjects	93 649	35 175 212
TOTAL ARTICLES	149 277	57 455 406

Table 2.1: Overview of scraped Wikipedia pages

2.2.3 Sub-dataset of Dialogues

We used the website `https://www.opensubtitles.org/sk` to create this sub-dataset. For the creation of the sub-dataset, AWS cloud services (Amazon Web Services, 2025b), namely Lambda and S3 bucket, were used. "AWS Lambda is a compute service that runs code in response to events and automatically manages the compute resources, making it the fastest way to turn an idea into a modern, production, serverless applications." (Amazon Web Services, 2025c). AWS S3 bucket is a storage space for files (Amazon Web Services, 2025a). The services were used because of the limitation of downloading files per IP address to 30 files per day. The update_function_code [1] function in AWS lambda allows you to update the code stored in the S3 bucket in .zip format at the same time when it is called, it creates a new IP address from which AWS Lambda can send requests. Therefore, the scraper architecture consisted of two Lambda systems and one S3 bucket serving as storage. The first AWS lambda served as a loop that sent input for a certain period of time (15 minutes) (https://www.opensubtitles.org/sk/search/sublanguageid-slo/movieyear-2024/offset-). The second AWS Lambda used the update_function_code function to refresh itself; the code was the same, but the IP address was updated, and the process could be repeated again. It sent 20 requests to different webpages containing subtitles and used the web-scraping method with the BeautifulSoup4 library. After saving the subtitles in text format to the S3 bucket, the second lambda process was terminated. After this time had elapsed, we ran the lambda again but with a different offset. In this way we eliminated another problem, namely the inability of the website to display more than 1300 headlines at once using a single url call.

[1] `https://docs.aws.amazon.com/lambda/latest/api/API_UpdateFunctionCode.html`

The last code execution was on 2024-07-26. On that date, there were 29 852 movies and series with Slovak subtitles on the site (`https://www.opensubtitles.org/sk/search/sublanguageid-slo`). After removing copies, incorrectly processed files, subtitles in a foreign language, applying preprocessing, and non-subtitle related sentences (translator's name, advertisement, etc.), we were left with 23 789 text files with a word count of 88 313 765. Due to the large amount of data, we took files that are larger in size than half of the total subtitle file to get 53 million words that fit the required word count.

2.2.4 Sub-dataset of Literature

We used 4 websites containing freely available books in Slovak language: (`https://www.zones.sk/`, `https://eknizky.sk/`, `https://greenie.elist.sk/`, `https://www.1000knih.sk/`). Web-Scraping of these websites was done differently, because we did not scrape the content of the page but we used web-crawling to download pdf documents but most of the time we used manual selection The resources from `https://eknizky.sk/` and `https://www.1000knih.sk/` were not available for download through our web-crawling or web-scraping tools, so we chose Manual selection, randomly selecting books that were in Slovak and were not poetry. Since most of the books were duplicates, we removed duplicate books due to the common name of the books before preprocessing. We then removed lines, using the delete_whole_line function, that contained information about the book such as author name, publisher, isbn, ean; single-volume books often contain a large number of dots, so we decided to remove lines containing 4 or more dots in a row. Furthermore, we removed the first page of each book that had text in it, as there was often a lot of text that was irrelevant to our pre-training. We then used the pymupdf library, which offered the ability to assess where paragraphs of text were located on a given page. Provided that all the paragraphs of text were on the same plane and the first and last paragraphs were in different positions, it could be assumed that it was a footer or header. By this principle, we tried to partially purify individual pages from footer and header.

Web pages	**Number scraped books**	**Number of words**
https://www.zones.sk/	6	246 362
https://eknizky.sk/	210	5 628 366
https://greenie.elist.sk/	22	423 527
https://www.1000knih.sk/	44	1 383 083
TOTAL BOOKS	272	7 681 338

Table 2.2: Overview of scraped book sources

2.2.5 Sub-dataset of Fairy tales

In creating the sub-dataset with fairytales, we used 7 webpages and random google search dealing with well-known fairytale authors:
(`https://www.sikovnamamina.sk/`, `https://www.rozpravkozem.sk/`, `https://www.zones.sk/studentske-prace/rozpravky/`, `https://rozpravky.online/`, `https://zlatyfond.sme.sk/`, `https://www.readmio.com/sk/uvod`, `https://svetrozpravok.sk/`) The process of creating this sub-dataset was very versatile because we worked individually with each source, using all the methods. Manual selection of fairytales was chosen due to the inability to retrieve the webpage using web-scraping and web-crawling methods, for example on the website `https://www.readmio.com/sk/uvod`, where we downloaded the pdf of the given tale. We can further mention the source `https://zlatyfond.sme.sk/`. Here we chose the manual selection method, where we collected links of subpages with stories, which we then scraped. The selection of fairy tales was based on several rules. We found the fairy tales on this page based on information about the author, or we read part of the text to verify that it was a fairy tale. These fairy tales were often not grammatically corrected, so we checked the grammar by finding typical and distinctive errors in the works (boly, robyly, etc.). Fairy tales were not always written in contemporary Slovak, so we also excluded fairy tales written in contemporary Slovak language. For other sources, we used webscraping and web-crawling methods.

Also, Due to the lack of specific text, we chose the text generation method. To create the fairytales, we used the gpt-4o language model and OpenAI Library. We created two prompts to generate the fairytales. The first prompt was used to create the topic of the fairy tale and the second prompt was used to create the fairy tale itself. A similar method of generating child-centered text can be found in other studies (Valentini et al. 2023, Schepens et al., 2023). However, we consider as a negative that in our case we cannot compare the accuracy of the result with the real vocabulary of children at a given age.

Sources	**Number of sources**	**Number of words in sources**
https://www.sikovnamamina.sk/	36	41 392
https://www.rozpravkozem.sk/	697	303 295
https://www.zones.sk/studentske-prace/rozpravky/	1 058	1 359 908
https://rozpravky.online/	87	43 636
https://www.readmio.com/sk	1 591	359 510
https://svetrozpravok.sk/	70	38 773
https://zlatyfond.sme.sk/	293	671 388
Downloaded books	30	509 365
Created fairytales	3 094	1 786 974
Total	**6 956**	**4 754 731**

Table 2.3: Overview of fairytale sources, number of items, and word counts

2.2.6 Sub-dataset of Educational content

In the creation of the sub-dataset we used the website `https://referaty.aktuality.sk/`. And used web-scraping, web-crawling methods, however we only used the beautifulsoup4 tool due to the limited number of sub-sites containing content such as `https://referaty.aktuality.sk/pedagogika`. So we omitted the scrapy tool. Another reason was the exclusion of foreign language related substrates `https://referaty.aktuality.sk/cudzie-jazyky`, which could contain a mix of foreign language and Slovak, which could lead to data retrieval in a foreign language. Subsequently, we applied pre-processing. We also tried to remove text regarding literature or web resources used using the delete_whole_line function. Due to the high word count, we discarded sources with fewer than 300 characters to reduce the number of possible badly scraped data and to reduce the number of sources. The resulting dataset contained 17 214 text sources and contained 14 954 348 words.

2.2.7 Sub-dataset of Child-directed speech

Datasets of child-directed speech already exist in different languages: the Child Language Data Ex- change System (CHILDES) (MacWhinney, 1998). No such dataset has been created within the Slovak language and we are not aware of an existing data source that is freely usable for our purposes. The CHILDES dataset exists in the Czech language [2], but after translation using LLM or the googletrans library (version 3.0.0) the translations contained frequent words in the Czech language and did not reach the quantity needed to create our dataset. Therefore, we decided to mechanically generate conversations between a child and a known person using LLM. We used the gpt-4o language model to generate the conversations. We chose GPT-4o based on the following features: persistent connection, ability to create this type of speech with good quantity.

A mechanical generation method to create a similar dataset has already been used in the context of creating a variation dataset that approximates the childes dataset. The Child-direct speech contains only the conversation from the mother's side, a given dataset is created by using a prompt that forces the LLM to repeat and reformulate the same content into a different form. Also, swap, add or change words but still keep the semantic intent (Haga et al., 2024). However, to better focus on the cognitive aspect of the conversation between the familiar person and the child, we decided to include features of the conversation between the familiar person and the child (We refer to the familiar person as a person in the child's close social circle, such as a parent, sibling, or guardian). According to the Usage-Based Theory of Language Acquisition

[2] https://childes.talkbank.org/access/Slavic/

by Tomasello (1992). Communication must take place in an activity or game where the child first passively and then actively participates in the events of the world. In a given game or interaction with a child, a familiar person assists in the proper development of language by repeating mispronounced words or describing the environment (Rowe and Snow, 2020).

Therefore, we decided to create 2 prompts in slovak language, the first prompt creates certain situation in which a child may encounter a familiar person and a conversation may take place between them, which will serve as a basis for creating a conversation. The second prompt adopted the given topic and other relevant information such as the average number of words spoken by the child during a specific age, of the child or specific age. Subsequently, during prompt engineering, we added queries about removing the greeting and removing the redundant text, for the sake of getting a better conversation. Each line of conversation was marked with start letter of person who speak. Since only the second-person conversation and not the child's will be used for pre-training, we did not focus on the linguistic correctness of the child's responses and removed lines starting with 'D:'. For more info see (AppendixB). Our resulting dataset consisted of 4 groups of conversations that were created based on the settings of the child's age and the number of words used in a sentence by the child at that age. Because of the recurrent topics and the generation of topics that children are unable to perform in the period below two years, we have omitted the period of one year.

The resulting text was saved as .json list with dictionaires, where the result of the first prompt was saved as the conversation source: Dictionary key: 'url'. The result of the second prompt as dictionary key: 'page'.

Age	**Number of conversations**	**Number of words in conversations**
2 years old	9 191	478 920
3 years old	7 764	477 217
4 years old	5 433	361 184
5 years old	7 688	415 297

Table 2.4: Number of conversations and words in conversations by age group

2.3 Application of CL solutions

Within our thesis, we will look at two ways of organizing the data The CL metrics application will be composed of some of the metrics already created within the given BabyLM challenge, plus new metrics will be created based on the literature collected. The metrics deal with two problems, masking words for pre-training the language model and the actual ordering of parts of the text from simplest to most complex. Separate evaluations will be created for the two problems. For word masking, this means evaluating each word that we consider important for masking based on the criteria, and then all tokens that the word contains will be masked. For text ranking, we will create several combinations of text, such as ranking each group of sub-datasets and then the given resources in the sub-dataset. For example, ranking the sub-datasets according to each metric. In the context of creating models, we will apply a certain method without applying any other method except masking, we will explain the reason in the masking words section.

2.3.1 Dataset layout strategies

In our research, we can divide the CL criteria into two groups, namely the group based on the linguistic part of the evaluation, which deals with the morphological, lexical and syntactic evaluations of the text, and the frequency part of the evaluation, which deals with the evaluation of the frequency of individual tokens, words and bi-grams. Due to computational complexity, the evaluation of the semantic component of the text has been omitted. The application of different solutions can be found in Babylm challenge (Warstadt et al., 2023b), which we also used in our research.

However, some linguistic complexity metrics were not successful, a study focusing on morphological and lexical complexity of words where the authors focused on Type-/Token Ratio, Punctuation density, Mean word length, mean and max rarity of words metrics proved to be unreliable for overcoming the random ordering of text (Edman and Bylinina, 2023). Another study by Bunzeck and Zarrieß (2023) used similar complexity metrics to rank text such as average word length, utterance length (the count of lexical tokens in the sequence) or average word frequency. However, again these proved to be unsuccessful against random text ranking. At the same time, a study by Agrawal and Singh (2023) tested the complexity of the datasets and their effectiveness for pre-training language models. The authors used similar metrics as in the previous study to evaluate complexity, while adding a Text Similarity and readability evaluation. The results of this study show that it is the models trained on more complex texts from the subsets of the aforementioned metrics that showed better performance in the downstream tasks.

Frequency complexity metrics can also be found in the Warstadt et al. (2023b). In a study of re-training the GPT-2 model on the babylm-challenge dataset, they used the average sentence frequency metric, not only to rank the text but also to remove text. For text removal, it was shown that if lines with low number of high frequency words but no semantic meaning were removed from the model, it improved the performance of the model on the BLIMP tuple. In the given study, the authors performed a total of 18 experiments when each dataset was ranked by semantic similarity and average sentence rarity. The experiments showed a positive relationship between the performance improvement on the BLIMP task and the given metrics (Borazjanizadeh, 2023). Another study first sorted the text by complexity. The authors organized the available datasets by whether the text in question was hovered or typed text. Hover text was considered "easier" and typed text was considered "harder". Only then did the authors apply different metrics to rank the text. One of them was token frequency, which was estimated from the token order in the BPE tokenizer. The results show that the given proposal for text ranking did not help to improve the results in the BLIMP task (Martinez et al., 2023).

As mentioned in the chapter on the differences between English and Slovak, Slovak is significantly more complex in terms of linguistic complexity. The consequences can be seen in the tokenization of text, which may differ based on the diversity of the language. A study examining text tokenization in 108 languages found that the ChatGPT-3.5 model required 2.13 times more tokens to tokenize English text than it did to tokenize English text (Asprovska and Hunter, 2024). From which we can conclude that the metrics of linguistic complexity or frequency complexity may have a greater impact on the ranking of text in Slovak, because more complex words that can be tagged with one token in English can be tagged with multiple tokens in Slovak, which may make the same word more difficult to predict. In terms of frequency, we can capture specific words in the corpus related to a specific topic. Therefore, despite the failure of linguistic metrics in the English language, we decided to apply morphological and lexical complexity of words, because of its possible greater effectiveness on the Slovak language. Regarding the frequency metrics, a study (Borazjanizadeh, 2023) demonstrated features capable of constructing a CL that outperforms random text ordering, but we also applied new frequency-based metrics for better text analysis.

The evaluation was done within a single source so as not to divide the context of the sentences. To evaluate linguistic complexity, we applied metrics:

Gramma complexity:

Average word length: The average word length is measured as the number of characters divided by the number of words in a given source. This metric indicates the lexical intensity of the resource, where longer words can reduce the readability of the text.

$$\textit{Average Word Length} = \frac{\text{Number of Characters}}{\text{Number of Words}} \tag{2.1}$$

Sylabelle/word ratio: The syllable/word ratio is measured as the number of syllables divided by the number of words. A given metric indicates the morphological complexity of the source, where a higher proportion of syllables per word may indicate a more complex word structure of the text.

$$\textit{Syllable/Word Ratio} = \frac{\text{Number of Syllables}}{\text{Number of Words}} \tag{2.2}$$

Conjunction ratio: Conjunction ratio is measured as the number of conjunctions divided by the number of words. In our implementation, we focused only on non-bending one-word conjunctions. The number of conjunctions was 59; a list of conjunctions can be found in Appendix A. Conjunctions have the task of linking sentence constructions, which can create large sentence constructions and thus increase the syntactic complexity of the sentence (Dvonč et al., 1966).

$$\textit{Conjunction Ratio} = \frac{\text{Number of Conjunctions}}{\text{Number of Words}} \tag{2.3}$$

Preposition ratio: The preposition ratio is measured as the number of prepositions divided by the number of words. In our implementation, we focused only on initial prepositions. The number of prepositions was 44; a list of prepositions can be found in Appendix A. Prepositions have the task of forming relations to flexible word types such as nouns or adjectives. At the same time, however, prepositions determine the case of the word they stand in front of (Dvonč et al., 1966). Inflected nouns may contain more tokens than nouns in the base form, so their presence may increase the morphological complexity of words.

$$\textit{Preposition Ratio} = \frac{\text{Number of Prepositions}}{\text{Number of Words}} \tag{2.4}$$

Punctuation density: punctuation density is measured as the number of punctuation marks divided by the number of words. For the calculation, we used the regex library to limit multiple punctuation marks after a line or dots appearing next to numbers such as in dates or numbering. A low number of punctuation marks and a high number of words can mean a complex sentence structure.

$$\textit{Punctuation Density} = -1 \times \frac{\text{Number of punctuation marks}}{\text{Number of words}} \tag{2.5}$$

Frequency complexity:

Prior to actual data sorting, we extracted the frequencies of individual tokens, words and bi-grams by splitting the words using the .split() function and removing non-alphabetical signs where appropriate, but we did not remove capital letters.

Average word frequency: The average word frequency is measured as the average of the individual word frequencies divided by the number of words in a given resource.

$$\textit{Average word frequency} = -1 \times \frac{\sum \text{word frequencies}}{\text{Number of all words}} \tag{2.6}$$

Average token frequency: The average token frequency is measured as the average of the individual token frequencies divided by the number of words in a given resource:

$$\textit{Average token frequency} = -1 \times \frac{\sum \text{frequency of individual tokens}}{\text{Number of all words}} \tag{2.7}$$

Average bi-gram frequency: The average bi-gram frequency is measured as the average of the individual bigram frequencies divided by the number of words in a given source:

$$\textit{average bi-gram frequency} = -1 \times \frac{\sum \text{frequency of individual bigrams}}{\text{Number of all words}} \tag{2.8}$$

The evaluation was done within a single sample (text from a single source) so as not to divide the context. The resulting ranking was created from the simplest to the most complex data sources. In order to properly measure the given metrics (lower rankings== simpler sentences), we had to rescale the frequency and punctuation density metrics (metric= -1*metric). The given metrics were then normalized using min-max normalization within their sub-dataset, and then the sum of the individual metrics together was performed. The result was used to rank the text from simplest to most complex.

2.3.2 Masking strategies

As we mentioned in the pre-training and finetuning MLM subsection, word masking is needed for language acquisition. Within the available literature, we can find several sources that show that non-random masking can significantly help to gain better results in test metrics than models masked on random tokens. One example is the change in the amount of masked words in the word masking process (from a high percentage of masked text to a low percentage) and the weighted masking that up-weighted non-functional words. Both masking methods showed improvement over random masking in downstream task performance (Yang et al., 2022). Furthermore, we can mention the masking of words based on the occurrence of significant collocations in the text (Levine et al., 2020). Several improvements in the BabyLM Challenge have also focused on token masking. An example is the masking of specific words essential to the BLIMP task. Choosing the correct grammatically spelled sentence in the Blimp task, consists in the interchangeability of words such as 'that' and 'what'. Therefore, specific words in different contexts have been masked. However Targeted MLM does not systematically improve performance, except for two BLiMP tasks out of 12.

We drew on statistical learning theory to construct LMs based on children's cognitive development and applied word masking based on the frequency of a single word throughout the text. We chose to apply word masking instead of tokens based on positive results in other studies (Wilf et al., 2023). At the same time, from a cognitive perspective, we perceive word masking as more akin to human perception. To compare the effect of frequency-based masking, we created a masking of the least frequent words and the most frequent words, where we masked the word by the number of tokens it contained. If multiple words had the highest or lowest frequency we randomly selected a word among these words. To mimic the learning method, word frequency was counted simultaneously with text masking.

Considering the comparison of the model with normal masking in normal pre-training, we investigated how many masks were created after applying the DataCollatorForLanguageModeling class on our text, which creates the masked text during training. We tried this experiment 4 times and the average number of masks created was 2 658 013 for the text we used to pre-train. Next, we experimented with how often we create masks from a word to get the same number of masks as in our first experiment. From the result of our experimentation, we got 7 words. The masking method was set to check the frequency of 7 words and then choose a word with the highest or lowest frequency to set mask token. The generated masked text targeting the most frequent words contained 2 232 040 masks and the masked text targeting the least frequent words contained 2 579 958 masks.

2.4 Creation of architecture and pre-training

The architecture of the model itself will be based on the results of the BabyLM challenge (Warstadt et al., 2023b). The given hyper-parameters of the models were tested on a strict-small task, which consisted of a smaller version of the dataset (10 million words) (Cagatan, 2023; Proskurina et al., 2023). Based on hyper-optimization of the parameters using the optuna library created two models encoder (4-layers, 8 attention heads) and decoder (6-layers, 12 attention heads) (Proskurina et al., 2023). They demonstrated a 1:2 ratio between feed-forward layers and attention heads as important. Therefore, our model will have 6 layers and 12 attention heads. We decided to take the other parameters for LM pre-training from the study Proskurina et al. (2023), except for the number of epochs, which was not included as a hyperparameter in the hyper-optimization. However, both studies pointed to lower max sequence,and it is 128 length as a suitable parameter for the given models (Cagatan, 2023; Proskurina et al., 2023). Even lower text batches of 64 or 32 were tested, which were also shown in the studies to be a factor for performance improvement Xiao and Zhu (2023); Cagatan (2023) but only at lower numbers of layers and attention heads Cagatan (2023). When comparing our model size with the tested models in the studies and the best parameters from individual pre-training, we decided to choose 15% masking percentage and 7 epochs for pre-training (Cagatan, 2023). During selection of tokenizer parameters, we slightly adjusted the parameters to the Slovak language compared to the results of. The given studies used a Vocabulary size of 40 000 and 30 000 (Edman and Bylinina, 2023; Opper et al., 2023) For more info see (AppendixD) As mentioned in the Dataset layout strategies section, Slovak language needs a larger number of tokens, so we increased this number to 60 000 words. The tokenizer type chosen was Bytelevel tokenizer.

We used the python programming language (version 3.8.10) transformers (version 4.46.3), tokenizers (version 0.20.3), datasets (version 3.1.0) libraries to train the models. Before starting the training, we defined what dataset we will draw from to create the correct tokenizer. Each trained tokenizer contained two files, namely the source text on which the individual tokenizer was trained (text_tokenizer.txt) and a json file that contained the id and the tokens belonging to the id (tokenizer.json). We invoked a given tokenizer using the Tokenizer class and the from_tokenizer function, which was then wrapped in the PreTrainedTokenizerFast class from the Transfomrers library, which allows other classes in the Transformers library to work with a given tokenizer. We then used the BertConfig class as a configuration class to set the various model parameters mentioned above. For training the model we used class BertForMaskedLM, as mentioned in the chapter on word masking, to create masked tokens in the text for proper pre-training we used class DataCollatorForLanguageModeling, when we set the percentage of masked text to 15%. To create the models, we used two graphics cards

namely NVIDIA GeForce RTX 3090 and NVIDIA GeForce GTX 1080. The individual settings of the other hyperparameters can be seen in the appendices.

To prepare the data, we chose a specific pre-processing, where we tokenized the text and divided it into parts containing 128 tokens, using a specific Chinese character that is not present in the Slovak text and split the text into batches. The reason was to reduce the number of padding tokens and thus we reduced the training time. To create a ratio of 80% training, 10% test, and 10% validation, we divided the text into ten parts and divided each part into an 8:1:1 ratio to ensure equal complexity of the training, test, and validation (evaluation) samples. We set the training parameters using the class TrainingArguments from the transformers library. However, due to the different text, we consider the testing results irrelevant and will not report them.

2.5 Model testing

To test the performance of the models, we selected two evaluation tasks. In the Babylm challenge, we used specific evaluation tasks such as BLIMP, which tests linguistics-related features such as understanding grammar or syntax. However, this evaluation task does not exist in Slovak. Within the Babylm challenge they used several other evaluation tasks. To get closer to this challenge, we will use 2 tasks from this challenge. Hence we will use sentiment analysis (SA), question answering(QA) tasks to highlight several features of the model. Both tasks point to different capabilities of the model, the QA task focuses on processing the text and producing an answer, which involves a more complex analysis of the relationships between words (Farea et al., 2022) and the SA task focuses more on a specific combination of words that produces a positive or negative sentiment.

(Pecar et al., 2019) The dataset structure for the SA task (DGurgurov/slovak_sa (Pecar, 2019)) consisted of 2 functions, namely text and label, where the label denoted the individual sentiment (0= negative, 1= positive) and the text function contained the analyzed text. A given dataset contains 3 560 samples in the train part, 522 samples in the validation part and 1 042 samples in the test part. For more detailed research, we determine up to 4 outcome metrics: Accuracy, Precision, Recall, F1-score. Accuracy is the ratio of the number of correct predictions to the total number of predictions made. Accuracy is the ratio of the number of correct positive predictions to the total number of positive predictions made. Recall is the ratio of the number of correct positive predictions to the total number of actual positive cases. The F1-score is the harmonic mean of accuracy and recall, thus equalizing the two metrics in a single score.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.9}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.10}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.11}$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.12}$$

Where:

- TP = True Positives
- TN = True Negatives
- FP = False Positives

- FN = False Negatives

The structure of the dataset used for the QA task (TUKE-DeutscheTelekom/squad-sk (Hládek et al., 2023)) consists of 5 features. Id, title, context, question, answers. Feature answers has two forms a to list of possible signs, which can be anwers in string data type and position tokens in context window. To better test the model, we removed questions with empty answers to force the model to produce an answer. After modifications, our final dataset contained 74 635 samples in train part, 5 729 samples in validation part, 8 292 samples in test part. Next important parameters for QA task was *n_best* and *max_answer_length* which post-process the answer from a LM. n_best parameter set from how many answers can be found right solution, which means we took the 20 most probable answers from model and we checked them with the right answer. The parameter max_answer_length sets how many tokens are allowed to avoid errors which focus on whole text and long answers. After several experiments, we put it on n_best = 20 max_answer_length = 50. To answer the questions, it is important to understand the context from which the model draws that answers the question and then create a specific answer therefore we will evaluate the correctness of the answer created [3]. We will use the F1 score and exact match as performance measurement metrics using the library evaluation (version 0.4.3). The F1 score computes the average of token-level precision and recall tokens between each prediction and the best reference response. The exact match score measures the agreement of predictions with at least one reference response.

$$EM = \frac{\text{Number of exact matches}}{\text{Total number of queries}} \tag{2.13}$$

$$Precision = \frac{\text{Number of matching tokens}}{\text{Number of predicted tokens}} \tag{2.14}$$

The resulting value will consist of the average of the individual runes. For all the evaluation tasks mentioned above, we used datasets from the dataset library, since the individual models will be fine-tuned for each task. We used the Transformers library to implement the fine-tuning in the same way as for training our models. For each task, we customized the final model layer, which modifies the model result, using the classes AutoModelForQuestionAnswering for the QA task and AutoModelForSequenceClassification for the SA task. Each model will be tested 9 times with 2 different hyper-parameters (3 different learning rates and 3 different epochs). For the SA task we chose the parameters: epochs = [5,7,10], learning rate= [5e-5,3e-5,1e-5]. However, testing the QA task takes longer, so we reduced the parameters, namely the number of epochs, to [3,5,7].

[3] `https://huggingface.co/docs/transformers/tasks/question_answering`

Chapter 3

Results

Our thesis focused on creating a dataset and testing CL metrics based on linguistic and frequency complexity. For the purpose of testing these metrics in practice, we created a small LM (Strict-model) of around 10 million words. We created 7 models, each with a different text ordering or applied stacked masking to test whether text ordering or mask application improved the model in the evaluation baskets, and another two to test the CL capability in preprocessing.

We can divide the created models into three groups according to three criteria. How the data will be sorted (no data sorted, sub-datasets sorted, full text sorted), what CL metrics will be applied to sort the text (frequency metrics only, grammar metrics only, both metrics), and what kind of masking will be applied (without specific masking, masking with max frequency, masking with min frequency):

Application of specific ordering:

1. Full ordering, without ordering, without specific masking
2. Files ordering, both metric groups, without specific masking
3. Full ordering, both metric groups, without specific masking

Application of group metrics:

4. Full ordering, only language group metric, without specific masking
5. Full ordering, only frequency group metric, without specific masking

Application of masking strategies:

6. Full ordering, both metric groups, masking with max frequency
7. Full ordering, both metric groups, masking with min frequency

For clarity in the results, we will provide only parameter that is important for comparison (e.g., sort type, CL metric group, masking method), and at the same time the models will be numbered; this numbering corresponds to the specific detailed setting from the above list.

3.1 Application of CL solutions

To evaluate the performance of each group of CL metrics, we compare 4 models. Namely, applying groups of metrics alone, groups of metrics together, and without applying our CL metrics.

Models	Exact Match (%)	F1 Score (%)
1. without sorting	1.385 (0.383)	**6.590 (0.648)**
3. both groups of metrics	1.334 (0.290)	6.435 (0.751)
4. language group	**1.391 (0.369)**	6.449 (1.200)
5. frequency group	1.222 (0.367)	6.451 (0.744)

Table 3.1: Results of QA task for application of CL metrics

Configuration	Loss	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
1. without ordering	**0.320 (0.106)**	**90.382 (2.653)**	**85.446 (8.128)**	**90.382 (2.653)**	**87.738 (5.544)**
3. both metric groups	0.321 (0.106)	90.158 (2.534)	85.141 (7.898)	90.158 (2.534)	87.427 (5.335)
4. language metric group	0.322 (0.103)	90.233 (2.590)	85.414 (8.101)	90.233 (2.590)	87.660 (5.485)
5. frequency metric group	0.339 (0.096)	89.657 (2.344)	83.506 (7.807)	89.657 (2.344)	86.359 (5.226)

Table 3.2: Results of SA task for application of CL metrics

From the results, we can observe that we were not able to improve the model performance in our tasks using CL metrics. The model without using CL metrics achieved the lowest loss and highest accuracy and F1 scores in the SA task and also in the Exact Match and F1 scores in the QA task. Even the frequency-based metrics achieved lower scores in both of our tasks on the other hand, we can observe a difference between language metrics and frequency metrics where the model trained on pure language metrics achieved similar results to the model without CL metrics indicating potentially higher relevance of language features over frequency features. This finding can be explained in two ways, namely from a linguistic perspective, where applying CL metrics to other languages may work differently since frequency metrics have been shown to be a positive factor for performance improvement in some studies with English (Borazjanizadeh, 2023). In the section Cross-linguistic differences between English and Slovak, we pointed out the different complexity of languages. Within the perceptual point of view, words in Slovak may form a larger number of token combinations and thus metrics based on language complexity may be more meaningful due to the higher morphological richness of Slovak, where individual words may have multiple shapes and variations and meanings. Which can lead to greater token diversity, affecting the way models process and learn language patterns. However, from an NLP perspective, individual tasks focus on a specific feature of the model. A study (Elgaar and Amiri,

2023) shows the application of different linguistic and frequency metrics to performance in evaluation tasks and demonstrated that a specific task needs a specific curriccullum for better results. Within the SA task, they pointed out to the gradual increasing variation of nouns and verbs as significant factors. Where linguistic metrics are more specialized than frequency metrics (long, complex words can be frequent in text). However, the given results confirm the results of studies in English (Bunzeck and Zarrieß, 2023; Martinez et al., 2023; Edman and Bylinina, 2023) on the inability of frequency and language metrics as suitable metrics for CL and improving the performance of the LM.

3.2 Text ordering methods

Within the ordering of individual sub-datasets, we can choose two ways of ordering. Order group sub-datasets and then order specific sources of data. The ordering of the data into groups results from the similar vocabulary and form of the text in the sources from each sub-dataset.

Model	**Exact Match (%)**	**F1 Score (%)**
1. without ordering	**1.3848 (0.3826)**	**6.5902 (0.6479)**
2. ordering of sub-datasets	1.3809 (0.4102)	6.5380 (0.6025)
3. ordering full data	1.3343 (0.2901)	6.4355 (0.7510)

Table 3.3: Results of QA task for data ordering strategies

Model	**Loss**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1 Score (%)**
1. without ordering	**0.320 (0.106)**	**90.382 (2.653)**	**85.446 (8.128)**	**90.382 (2.653)**	**87.738 (5.544)**
2. ordering by sub-datasets	0.335 (0.095)	89.742 (2.182)	84.562 (7.361)	89.742 (2.182)	86.879 (4.854)
3. ordering full data	0.321 (0.106)	90.158 (2.534)	85.141 (7.898)	90.158 (2.534)	87.427 (5.335)

Table 3.4: Results of SA task for data ordering strategies

The ordering of sub-data sets (Table 3.5) was by comparison of sub-data sets, creating the order (1 = the simplest, 6 = the most complex) in each metric, and make a summation.

	Avg word_rarity	**Avg tok**	**Avg bi_gram**	**Avg word_length**	**Syl./word**	**Conj.**	**Prep.**	**Punc. d.**	**Sum**
Child-directed speech	4	2	2	1	1	1	1	1	13
Dialogues	3	1	1	5	6	3	2	2	23
Literature	2	3	3	3	3	5	4	3	26
Fairytales	1	4	4	2	2	6	3	4	26
Educational content	5	6	6	4	4	4	5	5	39
Wiki	6	5	5	6	5	2	6	6	41

Table 3.5: Evaluation of sub-datasets by CL metrics to create ordering by sub-datasets

Our proposed data ranking methods do not show significant improvement in our evaluation metrics (Table 3.3; Table 3.4). The model without data sorting achieved the best results and in both tasks. On the other hand, the sub-datasets sorted as a single unit show worse performance in QA performance than the model with group sorting based on F1 score and exact match, and in turn, the model with group sorting performs worse in SA tasks based on F1 score, precision, accuracy, and loss. This may suggest the possibility of an effect of data ordering on context handling performance when the model was successively trained on different forms of context-specific text and thus the contextual coherence of the texts was not compromised. In Malkin et al. (2021), they demonstrate the absence of coherence and conent logic as a negative factor to handle longer-term dependencies between sentences and effective context work. As for SA tasks, this can be explained by the effect of variations in nouns and verbs, which by using metrics and ranking the whole dataset can effect this ranking (Elgaar and Amiri, 2023).

3.3 Analysis of masking techniques

For the masking analysis, we will compare 4 models namely models without masking, without masking but with the application of frequency and language metrics and then the application of masking, since it is a matter of counting the frequency during the masking process, we decided to apply our metrics to the text alignment.

Model	**Exact Match**	**F1 Score**
1. without ordering	1.384768 (0.38262)	**6.590166 (0.647894)**
3. with ordering but without masks	1.334342 (0.290131)	6.435486 (0.750956)
6. with ordering and min freq. mask	1.006575 (0.241234)	6.403664 (0.851644)
7. with ordering and max freq. mask	**1.398344 (0.194654)**	6.455729 (0.889105)

Table 3.6: Results of QA task for masking techniques

Model	**Loss**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1 Score (%)**
1. without ordering	**0.320 (0.106)**	**90.382 (2.653)**	85.446 (8.128)	**90.382 (2.653)**	87.738 (5.544)
3. with ordering but without masks	0.321 (0.106)	90.158 (2.534)	85.141 (7.898)	90.158 (2.534)	87.427 (5.335)
6. with ordering and min freq. mask	0.320 (0.088)	89.891 (1.787)	**88.613 (2.761)**	89.891 (1.787)	**88.556 (3.233)**
7. with ordering and max freq. mask	0.324 (0.095)	89.838 (1.759)	88.014 (4.607)	89.838 (1.759)	87.914 (3.886)

Table 3.7: Results of SA task for masking techniques

The created masking showed improvement only in the sentiment task (precision) however in the QA task the application had negative effects on the result. A significant

difference between minimum and maximum can be observed in the EM metric, where the model with masked words with minimum frequency outperformed the other models.

Most Frequent Words		Least Frequent Words	
Word	**Frequency**	**Word**	**Frequency**
sa	258273	ľúto	142
a	207047	istý	139
to	159399	rozhodovaní	130
je	131987	myslel	128
na	114932	myslela	121
v	104538	nepovedal	121
si	96317	vedel	120
som	68440	vedela	119
že	48899	nepovedala	118
s	38767	istá	114

Table 3.8: Top 10 Most Frequent and Least Frequent Words in each masking technique

From the 10 most masked words, we can observe that masking complex words decreased the results in the QA task when masking low frequency words, applied masking to words carrying context. At the same time, applying masking to low frequency words masked many more original words (458780) than masking high frequency words (17266).

From the results (Tables 3.6;3.7;3.8), we can conclude that masking low frequency words masked words with important contextual meaning at the same time these words contain a higher amount of tokens, masking them made it more difficult for the model to reconstruct the original meaning and thus negatively affected the performance in QA tasks where accurate text understanding is crucial. In a study Kang et al. (2020) pointed out the need for word masking within a domain, where the authors found that masked words do not aid learning of representations but the model learns representations of unmasked words to predict masked ones. Therefore, masking high-frequency words such as ('sa','a','to') may lead to better learning from context.

3.4 Metrics as preprocessing methods

Due to the sufficient size of the individual sub-datasets, we also tested the impact of CL metrics as preprocessing methods. In addition to the already established strict-model, where sources were randomly selected from the sub-datasets. We created 2 other models where we used the simplest and most complex text from the individual sub-datasets. For evaluation, we used a combination of frequency and grammar metrics, where we ranked the sub-datasets according to the metrics and then selected the text for pre-training the model.

Model	Exact Match	F1 Score
1. Random complexity	1.384768 (0.38262)	6.590166 (0.647894)
1. The simpliest complexity	**1.601986 (0.244648)**	**6.948253 (0.703147)**
1. The hardest complexity	1.283916 (0.360808)	6.066877 (1.25371)

Table 3.9: Results of QA task for complexity text

Model	Loss	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
1. Random complexity	**0.320 (0.106)**	**90.382 (2.653)**	85.446 (8.128)	**90.382 (2.653)**	**87.738 (5.544)**
1. The simplest complexity	0.326 (0.100)	90.051 (2.425)	85.050 (7.758)	90.051 (2.425)	87.359 (5.200)
1. The hardest complexity	0.321 (0.101)	90.318 (2.613)	**85.455 (8.128)**	90.318 (2.613)	87.726 (5.526)

Table 3.10: Results of SA task for complexity text

The following results demonstrate that using CL metrics as pre-processing has a positive effect on improving the performance of LM in the QA task, where the model pre-trained on the simplest text performed the best based on the Exact-match metric (Table 3.9). However, within the SA task we find the opposite effect where the model with the simpliest complexity achieved the smallest results according to Accuracy or Recall (Table 3.10).

After scanning the sub-datasets, we can talk about filtering out specific resources that may be ineffective for pre-training LM, such as in the wikipedia sub-dataset, where the most complex resources according to the metrics were:

https://sk.wikipedia.org/wiki/Redaktor:Teslaton/HashMapTplTest/HashMap
https://sk.wikipedia.org/wiki/Najdlhšie_slovenské_slovo

These resources contain long words or disjointed text, which can reduce the chance of the LM learning from a coherent text, hence it can reduce the ability to answer questions. Therefore, we can observe a slight improvement in the model with the simplest complexity, where the given model outperformed both models.

Chapter 4

Conclusion

The aim of the final thesis was to establish a cornerstone in the research of cognitively inspired models and to point out the possibilities of applying CL to LRLs such as Slovak language.

The first goal of this thesis was to create a Slovak version of the BabyLM challenge (Warstadt et al., 2023a). In order to open the way for future researchers creating LMs that are cognitively closer to the amount and variability of language acquired by humans. For future research, we created 6 sub-datasets replicating the BabyLM Challenge (Warstadt et al., 2023a). All resources were pre-processed, based on the specific conditions found in the Pre-processing of sub-datasets section, and then we approached each sub-dataset individually. This dataset can be found at `https://huggingface.co/datasets/ubokri/SlovakBabyLM`

Domain of Sub-Dataset	**Number of words**	**Sources**	**Number of Strict-model words**
Child-directed speech	1,7 mil	Text generation	470 000
Fairytales	4,7 mil	7 webpages + random books + Text generation	910 000
Dialogues	53,6 mil	`https://www.opensubtitles.org/sk`	4 000 000
Educational content	14,9 mil	`https://referaty.aktuality.sk/`	1 304 000
Wiki	22 mil	`https://sk.wikipedia.org/`	2 300 000
Books	7.6 mil	4 webpages	990 000
Total	**104,5 mil**		**9,974,000**

Table 4.1: Overview of sub-dataset domains, their size, and sources

The second goal was to conduct several experiments in CL, where we were inspired by cognitive and linguistic theories and drew on the linguistic foundations of the Slovak language. For our experiments, we used Vanilla CL to lay out the text (metrics based on frequency and linguistic complexity) and to create masks (masking based on word frequency). The constructed experiments demonstrated several findings that can help guide future research in CL. However, our CL designs did not improve the evaluation results in QA and SA tasks, so we decided not to create a model that copy human language acquisition from 100 million words, however we publish the created sub-datasets to serve in other CL research areas.

The results of our experiments corroborated the English language studies, where both sets of metrics showed no significant improvement on the QA and SA tasks over random text sorting (Martinez et al., 2023; Bunzeck and Zarrieß, 2023; Edman and Bylinina, 2023), but we demonstrated the superior performance of the language metrics over the frequency metrics. The positive result of sub-dataset ordering and the negative result of frequency word masking versus random masking but the positive effect of high frequency word masking, however, these relationships are demonstrated by very little difference from the random masking and source ordering model. Small differences can also be found in studies , where the proposed CL metrics improved or worsened the model performance on the evaluation tuples in percentages (Elgaar and Amiri, 2023; Bunzeck and Zarrieß, 2023). In addition, we can observe the effect of evaluation metrics for which ranking based on a different CL metric is also important, since testing model properties may involve searching for specific words or understanding context (Elgaar and Amiri, 2023). Nevertheless, our most significant differences are in the decimal places compared to research in English, can be considered as a much smaller effect. However, our results may be skewed by a combination of metrics, text processing, or failure to use more specific tasks.

Based on our results, we can conclude that the positive or negative effect of CL metrics in Slovak was much less significant than in English. According to Bengio et al. (2009), CL metrics need to gradually increase the amount of more useful information with increasing learning time, which may be more complicated in Slovak language due to its linguistic complexity, where the order produced by the metrics may be more difficult to determine compared to English.

Therefore, in future research in Slovak, we may focus on metrics that are more concerned with specific features of the Slovak language (suffixes and prefixes, timing, inflection). Further, we could focus on other aspects of the model's processing of text such as its perception of text, an example being the use of a BPE tokenizer adapted to Slovak morphology (Držík and Forgac, 2024), which would bring the model's processing of text closer to that of human processing of text. At the same time, we need to create new testing tasks that highlight different properties of models and create a combination of testing methods focused on language understanding to better understand what capabilities LM has.

4.1 Limitations

One of the fundamental points limiting the results was the generation of data using the LLM. We used the GPT-4o model to use the text in Slovak language, which met the qualities of price, reliability while maintaining connectivity, but on the other hand, after going through the text, we encountered grammatical errors, which we corrected during pre-processing, but due to the large content of the text, we did not have the capacity to manually correct a large amount of text.

Next, we can mention the quality of the datasets. As part of the comparison of the quality of datasets, we can mention the website we drew from. For a smaller number of sources, we manually exclude webpages containing grammatical errors such as `https://zlatyfond.sme.sk/` but Websites such as `https://referaty.aktuality.sk/` or `https://www.zones.sk/`, that may contain articles with a higher number of grammatical errors or articles with bullet points or non-alphabetical signs was not able to exclude due to the high number of webpages.

Within the limitations of computational metrics, we could not focus on the semantic part of the language. The results of studies with cosine similarity as an evaluator for CL metrics show that the factor would improve the evaluation, and hence we would get better results (Han and Myaeng, 2017; Borazjanizadeh, 2023). For instance, due to the computational complexity of metrics , we could not perform the Part of Speech evaluation or more deeper analysis of text.

As mentioned in the Conclusion, new tests need to be developed to evaluate LM from a linguistic perspective in order to verify the understanding of the language and its grammatical rules using LM.

Bibliography

Agrawal, A. and Singh, S. (2023). Corpus complexity matters in pretraining language models. In *Proceedings of The Fourth Workshop on Simple and Efficient Natural Language Processing (SustaiNLP)*, pages 257–263.

Allgower, E. L. and Georg, K. (2012). *Numerical continuation methods: an introduction*, volume 13. Springer Science & Business Media.

Amazon Web Services (2025a). Amazon S3. `https://aws.amazon.com/s3/` Accessed: 2025-04-12.

Amazon Web Services (2025b). Amazon Web Services. `https://aws.amazon.com/`. Accessed: 2025-04-12.

Amazon Web Services (2025c). AWS Lambda. `https://aws.amazon.com/pm/lambda/`. Accessed: 2025-04-12.

Armon-Lotem, S., Haman, E., de López, K. J., Smoczynska, M., Yatsushiro, K., Szczerbinski, M., and van der Lely, H. (2016). A large-scale cross-linguistic investigation of the acquisition of passive. *Language Acquisition*, 23(1):27–56.

Aslin, R. N., Saffran, J. R., and Newport, E. L. (1998). Computation of conditional probability statistics by 8-month-old infants. *Psychological science*, 9(4):321–324.

Asprovska, M. and Hunter, N. (2024). The tokenization problem: Understanding generative ai's computational language bias. *Ubiquity Proceedings*, 4(1).

Bandura, A. (1965). Vicarious processes: A case of no-trial learning. In *Advances in experimental social psychology*, volume 2, pages 1–55. Elsevier.

Barnett, M. (2024). regex. `https://pypi.org/project/regex/`. Accessed: 2024-04-11.

Becker, J., Wahle, J. P., Gipp, B., and Ruas, T. (2024). Text generation: A systematic literature review of tasks, evaluation, and challenges. *arXiv preprint arXiv:2405.15604*.

Beinborn, L. and Hollenstein, N. (2023). *Cognitive plausibility in natural language processing*. Springer.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Borazjanizadeh, N. (2023). Optimizing gpt-2 pretraining on babylm corpus with difficulty-based sentence reordering. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 356–365.

Buckner, C. and Garson, J. (2025). Connectionism. In Zalta, E. N. and Nodelman, U., editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Spring 2025 edition.

Bunzeck, B. and Zarrieß, S. (2023). Gpt-wee: How small can a small language model really get? In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 35–46.

Bérešová, J. (2016). Nominal tendencies in english and their slovak equivalents in contemporary literature. *Language, Individual & Society*, 10:37.

Cagatan, O. V. (2023). Toddlerberta: Exploiting babyberta for grammar learning and language understanding. *arXiv preprint arXiv:2308.16336*.

Charpentier, L. G. G. and Samuel, D. (2023). Not all layers are equally as important: Every layer counts bert. *arXiv preprint arXiv:2311.02265*.

Cheng, Z., Aralikatte, R., Porada, I., Spinoso-Di Piano, C., and Cheung, J. C. K. (2023). Mcgill babylm shared task submission: The effects of data formatting and structural biases. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 207–220.

Chomsky, N. (1980). A review of bf skinner's verbal behavior. *The Language and Thought Series*, pages 48–64.

Chomsky, N. (2014). *Aspects of the Theory of Syntax*. Number 11. MIT press.

Chowdhary, K. (2020). *Fundamentals of artificial intelligence*. Springer.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20:273–297.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (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.

Dolník, J. (2010). *Všeobecná jazykoveda. Opis a vysvetl'ovanie jazyka*. Vydavatel'stvo Slovenskej akadémie vied, Bratislava.

Držík, D. and Forgac, F. (2024). Slovak morphological tokenizer using the byte-pair encoding algorithm. *PeerJ Computer Science*, 10:e2465.

Dvonč, L., Ružička, J., et al. (1966). *Morfológia slovenského jazyka*. Ústav slovenského jazyka.

Edman, L. and Bylinina, L. (2023). Too much information: Keeping training simple for babylms. *arXiv preprint arXiv:2311.01955*.

Eisenstein, J. (2018). *Natural language processing*. MIT Press.

Elgaar, M. and Amiri, H. (2023). Ling-cl: Understanding nlp models through linguistic curricula. *arXiv preprint arXiv:2310.20121*.

Elman, J. L. (1993). Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99.

Etxaniz, J., Azkune, G., Soroa, A., de Lacalle, O. L., and Artetxe, M. (2023). Do multilingual language models think better in english? *arXiv preprint arXiv:2308.01223*.

Farea, A., Yang, Z., Duong, K., Perera, N., and Emmert-Streib, F. (2022). Evaluation of question answering systems: complexity of judging a natural language. *arXiv preprint arXiv:2209.12617*.

Feng, D., Gomes, C. P., and Selman, B. (2020). A novel automated curriculum strategy to solve hard sokoban planning instances. *Advances in Neural Information Processing Systems*, 33:3141–3152.

Friston, K. (2009). The free-energy principle: a rough guide to the brain? *Trends in cognitive sciences*, 13(7):293–301.

Futrell, R. and Levy, R. (2017). Noisy-context surprisal as a human sentence processing cost model. In *Proceedings of the 15th conference of the european chapter of the association for computational linguistics: Volume 1, long papers*, pages 688–698.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Haga, A., Fukatsu, A., Oba, M., Bisazza, A., and Oseki, Y. (2024). Babylm challenge: Exploring the effect of variation sets on language model training efficiency. *arXiv preprint arXiv:2411.09587*.

Halina, M., Rossano, F., and Tomasello, M. (2013). The ontogenetic ritualization of bonobo gestures. *Animal cognition*, 16:653–666.

Haman, E., Łuniewska, M., Hansen, P., Simonsen, H. G., Chiat, S., Bjekić, J., Blažienė, A., Chyl, K., Dabašinskienė, I., Engel de Abreu, P., et al. (2017). Noun and verb knowledge in monolingual preschool children across 17 languages: Data from cross-linguistic lexical tasks (litmus-clt). *Clinical linguistics & phonetics*, 31(11-12):818–843.

Han, S. and Myaeng, S.-H. (2017). Tree-structured curriculum learning based on semantic similarity of text. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 971–976. IEEE.

Hládek, D., Staš, J., Juhár, J., and Koctúr, T. (2023). Slovak dataset for multilingual question answering. *IEEE Access*, 11:32869–32881.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press.

Horsch, J. (2021). Typological profiling of english, spanish, german and slovak: A corpus-based approach. *Jazykovedný časopis*, 72(2):342–352.

James, G. (2023). *Introduction to Google Translate*. Gilad James Mystery School.

Jiang, L., Meng, D., Zhao, Q., Shan, S., and Hauptmann, A. (2015). Self-paced curriculum learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Jin, Y., Chandra, M., Verma, G., Hu, Y., De Choudhury, M., and Kumar, S. (2024). Better to ask in english: Cross-lingual evaluation of large language models for healthcare queries. In *Proceedings of the ACM on Web Conference 2024*, pages 2627–2638.

Jones, K. S. (1994). Natural language processing: a historical review. *Current issues in computational linguistics: in honour of Don Walker*, pages 3–16.

Joshi, P., Santy, S., Budhiraja, A., Bali, K., and Choudhury, M. (2020). The state and fate of linguistic diversity and inclusion in the nlp world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293. Association for Computational Linguistics.

Kang, M., Han, M., and Hwang, S. J. (2020). Neural mask generator: Learning to generate adaptive word maskings for language model adaptation. *arXiv preprint arXiv:2010.02705*.

Keller, F. (2010). Cognitively plausible models of human language processing. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 60–67.

Kesselová, J. (2014). Ako sa rodí pád. *Filozofická fakulta Prešovskej univerzity v Prešove*. Súčasť projektu VEGA 1/0129/12: Modelovanie rečového vývinu slovensky hovoriacich detí v ranom veku.

Kidd, E., Donnelly, S., and Christiansen, M. H. (2018). Individual differences in language acquisition and processing. *Trends in cognitive sciences*, 22(2):154–169.

Kirkpatrick, S., Gelatt Jr, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *science*, 220(4598):671–680.

Leonard, L. B., Hansson, K., Nettelbladt, U., and Deevy, P. (2004). Specific language impairment in children: A comparison of english and swedish. *Language Acquisition*, 12(3–4):219–246.

Levine, Y., Lenz, B., Lieber, O., Abend, O., Leyton-Brown, K., Tennenholtz, M., and Shoham, Y. (2020). Pmi-masking: Principled masking of correlated spans. *arXiv preprint arXiv:2010.01825*.

Lin, X. V., Mihaylov, T., Artetxe, M., Wang, T., Chen, S., Simig, D., Ott, M., Goyal, N., Bhosale, S., Du, J., et al. (2021). Few-shot learning with multilingual language models. *arXiv preprint arXiv:2112.10668*.

MacWhinney, B. (1998). The childes system. *Handbook of child language acquisition*, pages 457–494.

Mahowald, K., Ivanova, A. A., Blank, I. A., Kanwisher, N., Tenenbaum, J. B., and Fedorenko, E. (2024). Dissociating language and thought in large language models. *Trends in Cognitive Sciences*.

Malkin, N., Wang, Z., and Jojic, N. (2021). Coherence boosting: When your pretrained language model is not paying enough attention. *arXiv preprint arXiv:2110.08294*.

Marentette, P. and Nicoladis, E. (2012). Does ontogenetic ritualization explain early communicative gestures in human infants. *Developments in primate gesture research*, 6:33–54.

Marr, D. (2010). *Vision: A computational investigation into the human representation and processing of visual information*. MIT press.

Martinez, R. D., Goriely, Z., McGovern, H., Davis, C., Caines, A., Buttery, P., and Beinborn, L. (2023). Climb: Curriculum learning for infant-inspired model building. *arXiv preprint arXiv:2311.08886*.

Martins, P. H., Fernandes, P., Alves, J., Guerreiro, N. M., Rei, R., Alves, D. M., Pombal, J., Farajian, A., Faysse, M., Klimaszewski, M., et al. (2024). Eurollm: Multilingual language models for europe. *arXiv preprint arXiv:2409.16235*.

Maye, J., Werker, J. F., and Gerken, L. (2002). Infant sensitivity to distributional information can affect phonetic discrimination. *Cognition*, 82(3):B101–B111.

McNamara, T. P. (1992). Priming and constraints it places on theories of memory and retrieval. *Psychological Review*, 99(4):650.

Meta AI (2024). Llama 3 models. `https://ai.meta.com/llama/`. Accessed: 11.04.2025.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Misra, K., Ettinger, A., and Rayz, J. T. (2020). Exploring bert's sensitivity to lexical cues using tests from semantic priming. *arXiv preprint arXiv:2010.03010*.

Morerio, P., Cavazza, J., Volpi, R., Vidal, R., and Murino, V. (2017). Curriculum dropout. In *Proceedings of the IEEE international conference on computer vision*, pages 3544–3552.

Mroczkowski, R., Rybak, P., Wróblewska, A., and Gawlik, I. (2021). Herbert: Efficiently pretrained transformer-based language model for polish. *arXiv preprint arXiv:2105.01735*.

Nagy, W. E., Herman, P. A., and Anderson, R. C. (1985). Learning words from context. *Reading research quarterly*, pages 233–253.

Newman, R. S., Rowe, M. L., and Ratner, N. B. (2016). Input and uptake at 7 months predicts toddler vocabulary: The role of child-directed speech and infant processing skills in language development. *Journal of Child Language*, 43(5):1158–1173.

OpenAI (2022). Whisper. `https://openai.com/index/whisper/`. Accessed: 2025-04-11.

OpenAI (2023). Gpt-4 technical report. `https://openai.com/research/gpt-4` Accessed: 11.04.2025.

Opper, M., Morrison, J., and Siddharth, N. (2023). On the effect of curriculum learning with developmental data for grammar acquisition. *arXiv preprint arXiv:2311.00128*.

Ozsoy, M. G. (2024). Multilingual prompts in llm-based recommenders: Performance across languages. *arXiv preprint arXiv:2409.07604*.

Panocová, R. (2021). Basic concepts of morphology i. *Košice: Vydavatel'stvo Šafárik-Press*.

Pecar, S., Šimko, M., and Bielikova, M. (2019). Improving sentiment classification in slovak language. In *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, pages 114–119.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Perruchet, P. and Poulin-Charronnat, B. (2012). Word segmentation: Trading the (new, but poor) concept of statistical computation for the (old, but richer) associative approach. *Statistical learning and language acquisition*, (1):119–143.

Piaget, J. (2000). Piaget's theory of cognitive development. *Childhood cognitive development: The essential readings*, 2(7):33–47.

Pikuliak, M., Grivalskỳ, Š., Konôpka, M., Blšták, M., Tamajka, M., Bachratỳ, V., Šimko, M., Balážik, P., Trnka, M., and Uhlárik, F. (2021). Slovakbert: Slovak masked language model. *arXiv preprint arXiv:2109.15254*.

Poulin-Dubois, D. and Brosseau-Liard, P. (2016). The developmental origins of selective social learning. *Current directions in psychological science*, 25(1):60–64.

Proskurina, I., Metzler, G., and Velcin, J. (2023). Mini minds: Exploring bebeshka and zlata baby models. *arXiv preprint arXiv:2311.03216*.

Qin, L., Chen, Q., Zhou, Y., Chen, Z., Li, Y., Liao, L., Li, M., Che, W., and Yu, P. S. (2024). Multilingual large language model: A survey of resources, taxonomy and frontiers. *arXiv preprint arXiv:2404.04925*.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. (2023). Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR.

Radford, A. and Narasimhan, K. (2018). Improving language understanding by generative pre-training. `https://api.semanticscholar.org/CorpusID:49313245`.

Rakison, D. H., Lupyan, G., Oakes, L. M., and Walker-Andrews, A. S. (2008). Developing object concepts in infancy: An associative learning perspective. *Monographs of the Society for Research in Child Development*, pages i–127.

Rowe, M. L. and Snow, C. E. (2020). Analyzing input quality along three dimensions: interactive, linguistic, and conceptual. *Journal of child language*, 47(1).

Schuster, M. and Nakajima, K. (2012). Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE.

Sennrich, R. (2015). Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Sherman, B. E., Graves, K. N., and Turk-Browne, N. B. (2020). The prevalence and importance of statistical learning in human cognition and behavior. *Current opinion in behavioral sciences*, 32:15–20.

Siahaan, F. (2022). The critical period hypothesis of sla eric lenneberg's. *Journal of Applied Linguistics*, 2(1):40–45.

Sido, J., Pražák, O., Přibáň, P., Pašek, J., Seják, M., and Konopík, M. (2021). Czert–czech bert-like model for language representation. *arXiv preprint arXiv:2103.13031*.

Sinha, S., Garg, A., and Larochelle, H. (2020). Curriculum by smoothing. *Advances in Neural Information Processing Systems*, 33:21653–21664.

Skinner, B. F. (1958). Reinforcement today. *American Psychologist*, 13(3):94.

Snow, C. E. (1977). The development of conversation between mothers and babies. *Journal of Child Language*, 4(1):1–22.

Soviany, P., Ionescu, R. T., Rota, P., and Sebe, N. (2022). Curriculum learning: A survey. *International Journal of Computer Vision*, 130(6):1526–1565.

Spranger, M. and Steels, L. (2014). Discovering communication through ontogenetic ritualisation. In *4th International Conference on Development and Learning and on Epigenetic Robotics*, pages 14–19. IEEE.

Štefánik, J., Rusko, M., and Považanec, D. (1999). Frekvencia slov, grafém, hlások a ďalších elementov slovenského jazyka. *Jazykovedný časopis*, 50(2):81–93.

Suzuki, K. (2011). *Artificial Neural Networks*. IntechOpen, Rijeka.

Tamis-LeMonda, C. S. and Bornstein, M. H. (1994). Specificity in mother–toddler language-play relationships across the second year. *Developmental Psychology*, 30:283–292.

Thiessen, E. D., Kronstein, A. T., and Hufnagle, D. G. (2013). The extraction and integration framework: a two-process account of statistical learning. *Psychological bulletin*, 139(4):792.

Thomas, D. M. and Mathur, S. (2019). Data analysis by web scraping using python. In *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*, pages 450–454. IEEE.

Tomasello, M. (1992). *First verbs: A case study of early grammatical development*. Cambridge University Press.

Transformers (2024). Transformers documentation. `https://huggingface.co/docs/transformers/en/index` Accessed: 11.04.2025.

Tsvetkov, Y., Faruqui, M., Ling, W., MacWhinney, B., and Dyer, C. (2016). Learning the curriculum with bayesian optimization for task-specific word representation learning. *arXiv preprint arXiv:1605.03852*.

Urbániková, M. (2010). Lexical and semantic development of the basic vocabulary in english and slovak.

Vanden Broucke, S. and Baesens, B. (2018). From web scraping to web crawling. *Practical Web Scraping for Data Science: Best Practices and Examples with Python*, pages 155–172.

Vapnik, V. N. (1999). An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is All you Need. *arXiv (Cornell University)*, 30:5998–6008.

Wang, X., Chen, Y., and Zhu, W. (2021). A survey on curriculum learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):4555–4576.

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.

Warstadt, A. and Bowman, S. R. (2022). What artificial neural networks can tell us about human language acquisition. In *Algebraic structures in natural language*, pages 17–60. CRC Press.

Warstadt, A., Choshen, L., Mueller, A., Williams, A., Wilcox, E., and Zhuang, C. (2023a). Call for papers–the babylm challenge: Sample-efficient pretraining on a developmentally plausible corpus. *arXiv preprint arXiv:2301.11796*.

Warstadt, A., Mueller, A., Choshen, L., Wilcox, E., Zhuang, C., Ciro, J., Mosquera, R., Paranjabe, B., Williams, A., Linzen, T., et al. (2023b). Findings of the babylm challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*.

Waxman, S. R., Fu, X., Arunachalam, S., Leddon, E., Geraghty, K., and joo Song, H. (2013). Are nouns learned before verbs? infants provide insight into a long-standing debate. *Child Development Perspectives*, 7(3):155–159.

Weizenbaum, J. (1966). Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45.

Welnitzová, K. and Munková, D. (2021). Sentence-structure errors of machine translation into slovak. *Topics in Linguistics*, 22(1):78–92.

Wilf, A., Akter, S. N., Mathur, L., Liang, P. P., Mathew, S., Shou, M., Nyberg, E., and Morency, L.-P. (2023). Difference-masking: Choosing what to mask in continued pretraining. *arXiv preprint arXiv:2305.14577*.

Xiao, T. and Zhu, J. (2023). Introduction to transformers: an nlp perspective. *arXiv preprint arXiv:2311.17633*.

Yang, D., Zhang, Z., and Zhao, H. (2022). Learning better masking for better language model pre-training. *arXiv preprint arXiv:2208.10806*.

Yong, Z.-X., Menghini, C., and Bach, S. H. (2023). Low-resource languages jailbreak gpt-4. *arXiv preprint arXiv:2310.02446*.

Zheng, S., Liu, G., Suo, H., and Lei, Y. (2019). Autoencoder-based semi-supervised curriculum learning for out-of-domain speaker verification. *System*, 3:98.

Zhong, C., Cheng, F., Liu, Q., Jiang, J., Wan, Z., Chu, C., Murawaki, Y., and Kurohashi, S. (2024). Beyond english-centric llms: What language do multilingual language models think in? *arXiv preprint arXiv:2408.10811*.

Zhu, J., Wang, Z., Chen, J., Chen, Y.-P. P., and Jiang, Y.-G. (2022). Balanced contrastive learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6908–6917.

Appendix A: List of conjunctions and prepositions

Conjunctions = ["a","že", "i","keby", "aby","aj", "ak","keď", "keďže","ako","akoby","hoci", "ale", "alebo","lebo", "ani", "iba","tak", "takže", "teda", "totižto", "veď", "však", "žeby" "avšak", "až", "ba", "bár", "beztak", "buď","by", "či", "čím","čoby",'pričom' "čiže", "čo", "kým", "leda", "ledva", "len", "len čo", "totiž", "lenže","najprv", "nech", "než","nielen", "no", "nuž","pokiaľ", "pokým", "predsa", "preto", "pretože", "síce","sotva", "sťa", 'prípadne', 'poprípade', 'eventuálne']

Prepositions = ["bez", "cez", "do", "k", "medzi", "na", "nad", "o", "od", "okrem", "po", "pod", "pre", "pred", "pri", "proti", "s", "skrz", "u", "v", "z", "za", "ponad", "popod", "popred", "poza", "popri", "pomedzi", "znad", "spred", "zmedzi", "spod", "spopred", "sponad", "spopod", "spoza", "spopri", "spomedzi", 'zo','ku','voči','skrze','vo','so']

Appendix B: Prompts for Children's books and Child-directed speech

Children's books:
First prompt:
count = 200
"role": "user", "content": f"Vytvor mi {count} názvov rozprávok. Vráť len zoznam názvov rozprávok bez ďalšieho úvodu, číslovania alebo záverečných poznámok. Témy sa nesmú opakovať.", "role": "assistant", "content": "Si spisovateľ detských rozprávok."

Second promt: "role": "user","content": f"Vytvor rozprávku pre deti na tému:{topic}. Snaž sa využiť maximálny počet tokenov.", "role": "assistant", "content": "Si spisovateľ detských rozprávok."
Child-directed speech:
count = 200
family = 'Matkou'
age_word = 'dvojročným'
age = 2
average_number_words = 2
First prompt: "role": "user", "content": f"""Vytvor {count} situácii medzi {family} a dieťaťom, ktoré sa môžu vyskytnúť medzi {family } a {age_word } dieťaťom. Výsledok budú len dané situácie a nebudú sa opakovať. Príklad: Prebaľovanie. Dieťa nesmie byť súčasťou činnosti, ktorú je nemožné vykonať v danom roku života ({age } roky). """, "role": "assistant", "content": "Si schopný posúdiť lingvistické a motorické prejavy dieťaťa v rôznom veku."
Second prompt: "role": "user", "content": F"""Daj mi konverzáciu medzi {age_word} dieťaťom a {family } na tému:{topic}. Tvoj výsledok musí obsahovať len vytvorený dialóg, kde {family } bude označená ako {family[0].upper() }: a dieťa ako D. Správna komunikácia zo strany {family}: Komentovanie: Je dôležité opisovať, to čo sa deje v okolí. Opakovanie: Zdôrazňovať a opakovať veci, ktorým dieťa nerozumie a poskytnúť

možností na neustále opakovanie nových slov alebo viet Výslovnosť: Použi slová gramaticky správne. !!!!! Prispôsobivosť: family musi prispôsobiť reč aktuálnym záujmom a potrebám dieťaťa: vety sú krátke!!!!! Priemerný počet slov vo vete u dieťaťa: {average_number_words} Nezačínaj komunkáciu pozdravom"", "role": "assistant", "content": "Si schopný posúdiť lingvistické a motorické prejavy dieťaťa v rôznom veku."

Appendix C: Implementation preprocessing

```
import re
url_regex = re.compile(r'(?:((http|https):\/\/|www)(?:\w+|\.)(?:\S+|\.)?(?:\S+\.)?)|(?:.*?\.sk(?=[\s\p{P}\n]))')
email_regex= re.compile(r'(\S+)(\.)?(\w+)@(\w+)(\.)(\S+)')
punctuation_regex= re.compile(r'''([~          !\"#$%&'()*+,-.\/:;<=>?@[\\\]^_`{|}~])\1+''')
phone_regex = re.compile(r'\d{5}[-\s]\d{4}[-\s]\d{4}|\d{3}/\s\d{3}\s\d{2}\s\d{2}')
def replace_urls(text):
    text=url_regex.sub('<URL>',text)
    return text
def replace_emails(text):
    text=email_regex.sub('<EMAIL>',text)
    return text
def replace_phones(text):
    text=phone_regex.sub('<TEL>',text)
    return text
def delete_double_punctuation(text):
    text=punctuation_regex.sub(' ',text)
    return text
def delete_double_spaces(text):
    text=re.sub(r'\s{2,}',' ',text)
    return text
def delete_double_newline(text):
    text=re.sub(r'(\n{2,}|\s*\n\s*\n\s*)', '\n', text)
    return text
def delete_whole_line(target,text):
    text= re.sub(r'^.*' + re.escape(target) + r'.*$\n?', "", text,
       flags=re.MULTILINE)
    return text
def replace_string(input,output,text):
    text=re.sub(input,output,text)
    return text
```

Appendix D: Model settings for pre-training

BertConfig

vocab_size = 60000

hidden_size = 84

num_hidden_layers = 6

num_attention_heads = 12

intermediate_size = 1446

hidden_dropout_prob = 0.15

attention_probs_dropout_prob = 0.3

hidden_act = "gelu_new"

TrainingArguments

num_train_epochs = 7

per_device_train_batch_size = 32

per_device_eval_batch_size = 32

evaluation_strategy = "steps"

eval_steps = 1000

save_steps = 1000

logging_steps = 100

load_best_model_at_end = True

metric_for_best_model = "eval_loss"

bf16 = True