Evaluating Utility of Vocabulary to Language Learners

Leonard Paál
Hochschule Karlsruhe
Supervising professor: Jannik Strötgen
Additional supervision by: John Blake (University of Aizu)

December 2024

Abstract

[Abstract]

Contents

1 Introduction **5**
1.1 Motivation . . . 5
1.1.1 Role of Vocabulary in Language Acquisition . . . 5
1.1.2 Context-specific Language Learning . . . 5
1.1.3 Examples of Contexts and Words . . . 6
1.2 Challenges and Contributions . . . 7
1.2.1 Challenges/Research Questions . . . 7
1.2.2 Contributions . . . 7
1.3 Outline of Work . . . 7

2 Background **8**
2.1 Context of the Work . . . 8
2.1.1 Linguistic Motivation . . . 8
2.1.2 Statement of Goal . . . 9
2.2 Natural Language Processing . . . 11
2.2.1 Corpus . . . 11
2.2.2 NLP Task . . . 11
2.2.3 NLP Preprocessing: Tokenization . . . 12
2.3 Artificial Intelligence . . . 13
2.3.1 AI Models . . . 13
2.3.2 Transformer Model . . . 13
2.3.3 Summary . . . 14
2.4 Explainable AI . . . 14
2.4.1 Distinctions between XAI Methods . . . 14
2.5 Ranking Vocabulary Lists . . . 16
2.5.1 Expert Judgement: Reading Rockets Tier System . . . 16
2.5.2 Frequency-Based Methods . . . 17
2.5.3 Summary . . . 19

3 Approach **21**
3.1 Formal Problem Statement . . . 21
3.2 Experimental Setup: Measuring Word Utility as Ability Improvement in Humans . . . 23
3.3 Experimental Setup with AI Model . . . 24
3.4 Evaluating vs. Making Lists of Useful Vocabulary . . . 26
3.5 Generating Efficient Lists of Vocabulary . . . 27

4 Implementation **29**
4.1 Data Pipeline . . . 29
4.2 NLP Tasks and AI Models . . . 30
4.2.1 Desiderata . . . 30
4.2.2 Task Selection . . . 32
4.2.3 NLP Pre-Training Tasks Used by State-of-the-Art AI Models . 32
4.2.4 Tasks Considered . . . 33
4.2.5 Sentence Embedding Methods . . . 34
4.2.6 Data Required for Each NLP Task . . . 34
4.2.7 AI Models . . . 34
4.3 Corpora . . . 34
4.3.1 Desiderata . . . 34
4.3.2 Corpora Used . . . 36
4.4 XAI Methods . . . 39
4.4.1 Desiderata . . . 39
4.4.2 Model-Agnostic Approaches . . . 40
4.4.3 Model-Specific Methods . . . 41

5 Evaluation **43**
5.1 Evaluation Measures . . . 43
5.1.1 NLP Task Performance . . . 43
5.1.2 List Similarities . . . 43
5.2 Baselines . . . 46
5.2.1 Existing Lists . . . 46
5.2.2 Existing List Generation Methods . . . 47
5.2.3 LLM Prompts . . . 47
5.2.4 Sample text . . . 47
5.3 Results . . . 48
5.4 Discussion . . . 48

6 Outlook **49**

7 Appendix **50**
7.1 Abbreviations . . . 50

List of Figures

3.1 Visualization of vocabulary list efficiency: **Blue** represents the score with learning with an efficient list, **Red** represents the score learning from a less efficient list (i.e, less useful words at the top of the list) . . 23

4.1 The pipeline producing the OpenSubtitles parallel corpus 37

List of Tables

2.1 XAI methods used in this work 15

3.1 Correspondence of abstract concepts to parts of implementation. . . . 26

Chapter 1

Introduction

1.1 Motivation

1.1.1 Role of Vocabulary in Language Acquisition

Learning a second language involves many different skills, often categorized into listening, reading, speaking, and writing. Another categorization may be vocabulary, grammatical skills, the ability to understand known words in various accents, understanding language when spoken at a fast speed. One skill that is required for any of these if the knowledge of vocabulary in the target language. A person with basic grammatical skills but no vocabulary has no ability to express themselves or understand anything which they hear around them. On the other hand, a person familiar with rudimentary vocabulary but no grammatical knowledge may struggle with understanding complex sentences and sound unnatural when speaking, but can at least make sense of short phrases and express themselves. Thus, basic knowledge of vocabulary is clearly one of the most essential skills for using a language. This raises the question of which vocabulary should be learned first when starting out on the journey of language acquisition. This work attempts to contribute to answering this question.

ToDo: Mention various methods of finding useful vocabulary from viewpoint of learner: textbooks, apps, etc.

1.1.2 Context-specific Language Learning

Learners of languages are typically interested in one or more specific aspects of the language. There is no such thing as an unbiased corpus than can fit every use case. Decisions must be made how much focus is given to everyday conversation, academic writing, writing pertaining to business like job applications etc. Many textbooks group vocabulary by topic, with a new topic being introduced with each lesson that student should ideally be able to converse in after completing the lesson. However, this way of introducing vocabulary has several shortcomings:

- Students spend time learning specific terms about the topic in one lesson while not learning even general vocabulary in other topics until much later.
- Knowing the words from previous lessons becomes a prerequisite for the more advanced material, especially because the terms from earlier lessons are used

in example sentences for grammar. Thus, learners interested in learning the use of the language in one context will have a hard time of skipping earlier, less pertinent lessons to them.

(cut segment about frequency and TF-IDF here)

Thus, there is a need for further exploration as to how word utility can be calculated, using modern NLP methods involving artificial intelligence where necessary. Put more simply, this question may be reduced to:

What order or words, when learned, gives the learner the best set of words to understand and communicate in the language as quickly as possible?

1.1.3 Examples of Contexts and Words

Some examples for contexts that could be interesting to language learners include:

- Reading Wikipedia articles about a specific field (computer science, literature, biographies)
- Watching movies
- Travel to a country where the target language is spoken
- Doing business with a company from a country where the target language is spoken
- Cultural exploration (literature, religion)
- Finding friends from other countries

The different contexts for which learners might be motivated to learn a language differ in how easily corpora can be obtained about to mine patterns from. Movie subtitles and Wikipedia articles are easily obtained from sites such as opensubtitles.org and wikipedia.org. The words that might be relevant for travel are not as easily obtained: One might imagine an ideal scenario to collect data, in which a statistically relevant group of people travelling to the destination to be examined are randomly selected and equipped with microphones and cameras before the travel. During travel, one could record their conversations, conversations with people around them, and materials they attempt to read to navigate their journey such as train schedules, descriptions of tours, restaurant menus, street signs, etc. Lacking the funds to conduct an experiment for every possible language, this work is interested in finding a methodology to obtain data from readily available corpora and websites online that extracts relevant vocabulary from the source texts. Depending on the context, we can think about which of the following English words might be likely to appear frequently in the texts:

- Convert
- Cash
- Hug

- Dammit
- Y'all
- From
- Nineteen eighty-four
- Married

To examine a few examples: Words like "convert" occur frequently when looking at Wikipedia articles [1]. The word "cash" is likely to be useful for travelers, but in most other contexts, it would not be as relevant. "Y'all" is almost never used in formal writing but used abundantly in everyday speech in the southern United Stated of America and South Africa. "From", meanwhile, will be likely to be one of the most frequently used words regardless of context.

1.2 Challenges and Contributions

1.2.1 Challenges/Research Questions

[What is the problem to be solved? - Given a context, find vocabulary maximally useful for understanding texts in it] Criteria for good extraction method:

- Requires little manual effort to generate input data (ideally performable on freely available corpora)
- Requires little computational effort
- Outputs word utilities that align with human intuition

Inherent challenges of text-based language analysis include:

- ambiguity
- Words having multiple different meanings (polysemy). E.g., *can* can be a verb or a vessel.

1.2.2 Contributions

[Motivation: Useful vocabulary -¿ Need approach to evaluate utility of word] [How this work address the challenges: Simulate human trying to understand texts with AI] [Distinguish between passive and active utility]

1.3 Outline of Work

Chapter content: Explain the logical flow of the thesis chapter to chapter

[1] See "Wikipedia" corpus 2016, drawn from one million lines on https://wortschatz.uni-leipzig.de/en/download/English

Chapter 2

Background

This chapter gives context to estimating the utility of vocabulary. We first discuss how this work connects to research about vocabulary acquisition for second language learners in Section 2.1. After this, we define the exact aim of the work in Section 2.1.2. The following three sections explain the (partially overlapping) fields of Natural Language Processing, Machine Learning, and Explainable AI, tools from all of which will be employed in our word utility evaluation approach laid out in Chapter 3. Finally, we examine current approaches for selecting vocabulary for language learners in Section 2.5, to show what methods are currently employed and how they could be improved.

2.1 Context of the Work

This section contextualizes the contents of this work by introducing theoretical considerations by linguistic researchers on the point of vocabulary lists. This will serve to motivate our research, and make clear what it does and does not aim to achieve, which is explicitly laid out in the second half of the chapter.

2.1.1 Linguistic Motivation

The issue of which words to learn when setting out to acquire a language is not a new one. Every textbook on language which does not focus exclusively on grammar must address the issue, and proactive learners will doubtless find themselves looking for ways to optimize the return on their invested studying time.

The Routledge Handbook of Second Language Acquisition [16] gives three criteria for deciding the order in which words are taught to beginners:

1. Frequency
2. Usefulness
3. Easiness

The first criterion, frequency, is easy to justify: Learning words that appear with great frequency will allow the learner to understand the maximum percentage of words in texts, and should thus be among the most relevant. The *easiness*, or *learnability* of a word is defined as how easy it is for a learner to acquire the word.

The authors define it with respect to an individual learner since it is posited that the various linguistic backgrounds of learners will influence which words they can acquire with ease. Apart from learner-independent factors influencing the learnability of a word such as word length, whether the learner already knows a cognate of the word will certainly also have an impact:

We can imagine a native German speaker attempting to learn English: The word "internationalization" may not be a particularly frequent word in most contexts and thus not seem too important to teach, but the German will likely recognize the word as a cognate of the German equivalent "Internationalisierung", and acquire the word with ease, giving a justification for teaching the word early if we wish to impart the most language knowledge as quickly as possible.

Finally, there is the criterion of "*usefulness*". While it is distinguished from frequency, the authors do not define what constitutes usefulness, or how it might be measured.

In their 2019 paper [8], He and Godfroid pick up the above three criteria in order to group words into clusters which should help determine their priority in vocabulary learning. However, again it is not defined what constitutes usefulness and no way to measure it is suggested beyond "human intuition". In fact, it is put in contrast with frequency, which may be derived from corpus data, whereas usefulness cannot.

As the aim of this work is to find ways to evaluate word utility automatically, the following section will attempt to define utility and related concepts more carefully, in order to state the aim of this work more formally.

2.1.2 Statement of Goal

Chapter content: Statement of goal, definition of "utility", context

Generally speaking, this work addresses the problem of evaluating how useful it is to learn a given word in the target language of a language learner. This helps us in generating ordered lists of vocabulary to learn which aid the learner in gaining competency in their target language quickly. Since different language learners learn languages for different reasons [Missing citation about: Statistics on motivations of language learners], we attempt to take into account the learner's motivation to tailor vocabulary specifically to the individual.

To make this goal more specific, we define several concepts that help us speak more precisely about this goal and how it may be accomplished. These concepts are used throughout this work.

Linguistic Context

By a linguistic context or just *context*, a subset of situations is meant in which a person interacts with language. Examples of language contexts by topic are: News, football, crocheting.

Examples of language contexts by situation are: Everyday conversation, scientific articles, watching movies.

Contexts can be further subdivided into smaller contexts, or grouped across different lines: "News" could be subdivided into "international politics", "sports", "business", etc.

The concept of a language context is useful to a language learner since it enables them to prioritize learning vocabulary and grammar associated with the con-

text they are interested in: Someone who is learning Arabic to better understand middle-eastern politics will have little use for words which are mostly associated with football.

The context could even determine the complexity of sentences that a learner must be able to understand: Some contexts, like everyday conversation, can be navigated fairly well with short sentences, but political speeches often feature long and structurally complex sentences, with various relative clauses, multiple negations, etc. [Missing citation about: Complexity differing across contexts] Thus, learning for one or multiple specific contexts gives the learner a more concrete goal and thus better idea of what they must learn to achieve it.

In contemporary methods and tools for language learning, usually little emphasis is placed on learning for a specific context, or else only a small number of contexts is available [Missing citation about: Investigate current popular language tools. Classroom cannot cater to individual preferences and apps only allow some personalization].

Utility

In this work, the *utility* of a word is defined as the increase of language ability in a given context that a person gains by knowing a word versus not knowing it at all. Here, language ability refers to being able to understand more of texts they read in the given context, being able to speak about the subject, etc. It is easy to see that some words will be more useful than others given a context: In the context of international news, learning the word "war" will bring more understanding to the learner than "pear" or "polymorphism".

Proxy Task

Some of the concepts in the realm of language learning cannot be measured directly: This can be because they are psychological in nature ("understanding") or because they are too complex to be objectively measured by numbers ("language ability"). The above section defines word utility as an increase in "language ability", but to analyze this ability with computers, we must make be able to put a number on it, even though we lack both an agreed upon scale and unit of measurement.

To circumvent this issue, so-called *proxy tasks* will be used: Proxy tasks are tasks that test subjects can perform, where the test result (the *proxy measure*) is used to capture the abstract quantity under study [Missing citation about: Proxy task definition]. For example, as a proxy measure for a person's general spelling ability, we could make them choose from multiple spelling variants of a word, and use the accuracy with which they select the correct answers.

This work employs AI models solving NLP tasks as an economic and repeatable substitute for real humans interacting with language, and uses the NLP tasks as proxy tasks to make "language understanding", and thus "utility", measurable concepts that can be computationally maximized.

Summary

With the concepts above defined more precisely, we can state the goal more accurately, namely, creating computational methods which can find words that have the

maximum *utility* given a particular *language context* by means of *proxy tasks*.

With the words in cursive filled in, this translates to: Finding words that provide a language learner with the most language understanding in the specific situations they are interested in, requiring them to learn the least possible amount of words. Since the understanding of a learner cannot be directly measured, we use tasks that check the learner's understanding as an imperfect but necessary approximation.

The following sections will discuss the fields addressed in achieving this aim, and highlight relevant aspects from them that contribute in later chapters to conceptualizing and implementing a solution.

2.2 Natural Language Processing

While the aim of the work is to make vocabulary lists that serve human learners, the technical implementation of this necessarily will process human language with the use of computers. This domain is called *Natural Language Processing*, a field defined in The Handbook of Computational Linguistics and Natural Language Processing as the engineering domain of Computational Linguistics [2]. Natural Language Processing, often abbreviated as *NLP*, is both used in Computational Linguistics (the field concerned with analyzing natural language through language data) and as well as in practical applications such as Chatbots, Machine Translation, Speech Recognition etc. [10]. This work addresses both aspects, as it is an undertaking of Computational Linguistics through the analysis of how AI models perform typical **NLP tasks**.

2.2.1 Corpus

The typical way Computational Linguistics empirically analyzes language is through the use of **corpora**. A common definition of a corpus can be seen in [9] as "an electronically stored collection of samples of naturally occurring language", which "can be a test bed for hypotheses and can be used to add a quantitative dimension to many linguistic studies".

Recently, many large corpora have been compiled and are freely available to the public. Corpora differ from each other mostly in their source and method of compilation. Depending on their source, some corpora may serve as examples of language being used in a language context, such as news or movie subtitles.

In the implementation and evaluation of this work's approach to word utility estimation, several such context-specific corpora will be used, as they present a way to analyze how language (and more specifically, vocabulary) is used in the linguistic context of the corpus. ToDo: Quote previous work that uses corpora to model linguistic contexts for the purposes of language learning

2.2.2 NLP Task

This work attempts to make use of not only static data in the form of corpora, but also the interaction of AI models with those corpora in order to gain insights into word utility. This active part of Natural Language Processing consists of performing **NLP tasks**. Typical NLP tasks include [10]

- Sentiment Detection: Given a text, estimate the emotional state of the author.
- Masked Language Modeling: Given a text with a word blanked out, estimate what the word should be.
- Machine Translation: Given a text, translate it another language while preserving meaning as faithfully as possible.

Humans are not trained from childhood to do any one language "task": They can have conversations, answer questions about things they have seen, etc. In contrast, the interaction of computers is usually more rigidly defined by specific NLP tasks, and AI models are trained to perform one or several of these tasks.

For the purposes of this work, an *NLP task* is a function with a specific input and output format, where at least one of the two formats takes the form of natural language. NLP tasks will serve as a way for AI models to interact with natural language, enabling us to analyze how certain words influence this interaction; the exact approach is described in Chapter 3.

2.2.3 NLP Preprocessing: Tokenization

Language presents itself in continuous form in most situations: When listening to spoken language, it is not obvious where one word ends and another begins. Likewise, written texts we find online or in books are not necessarily subdivided into its semantic constituents. While words in the written English language are mostly separated by spaces, a writer may choose to create a new hyphenated word sequence on the spot as necessity demands.

Further complicating the issue of where to separate words is the fact that many non-European languages do not use spaces in their spelling (e.g. Japanese, Mandarin Chinese) or use spaces for a different purpose (separating syllables in Vietnamese, separating sentences in Thai). For these reasons, *tokenizers* are used in Natural Language Processing to divide continuous texts into individual words [10]. ToDo: Explain various types of tokenization: Sub-word, word etc.

Splitting continuous text into distinct words will be required for this work and will serve the following purposes:

- Making statistics from word counts (e.g., counting which words occur many times).
- Assigning estimated utility to words.
- Being able to remove individual words in text inputs to AI models, in order to test what effect removing a particular word has on the output.

In order to provide utility evaluation for words in a wide range of languages, this work will employ tokenizers which can handle language of different syntactic structures.

In recent years, major advances have been made in the field of Natural Language Processing through the use of AI models. As this work makes extensive use of AI models, the next section therefore briefly introduces the field of Artificial Intelligence.

2.3 Artificial Intelligence

Elaine Rich, in her 1983 work "Artificial Intelligence", defines Artificial Intelligence as "the study of how to make computers do things at which, at the moment, people are better" [25]. An introductory article by IBM describes it as "a technology that enables computers and machines to simulate human learning, comprehension, problem-solving, decision-making, creativity and autonomy"[1]. This is opposed to traditional algorithms which can only follow algorithms which have been explicitly programmed.

2.3.1 AI Models

The building block of Artificial Intelligence is called an **AI Model**, which is a model using training data to learn patterns, and then used on inputs that were not identically present in the training dataset.

AI models are used in this work to simulate an agent interacting with language. The (pre-trained) AI model is thought of as a test subject in possession of linguistic skills that can be studied. Through its training process, AI models learn patterns that are generalizable across the problem domain. If these patterns resemble the knowledge that humans gain when they learn a new ability, it may be possible to analyze the interaction of AI models with data so that a human learner can learn from their behavior, similar to how humans can observe experts in their domain and learn from their behavior. The field which provides these methods of analysis is Explainable AI, which is explained in Section 2.4.

The current de facto standard for recent AI models is the *Neural Network* architecture [10], which attempts to imitate the neural makeup of the human brain. Such models achieve state-of-the-art results on most NLP tasks [Missing citation about: neural networks performing NLP task], and all models used in this work are Neural Networks for this reason. However, within this architecture there exist further subtypes of AI models, the most important for this work is the **transformer model**.

2.3.2 Transformer Model

Since their invention in 2017 [29], *transformer* models have brought impressive gains in performance to many fields, including Natural Language Processing.

They are a particular type of deep neural network characterized by an attention layer before the fully connected neural layers. Said attention layer makes the model "focus" on important parts of the important, while "ignoring" less important ones. This helps the model find patterns in noisy input. Attention is one possible way to attribute importance to inputs, and thus utility to words in our case. This use will be discussed in Section 4.4.

[1] `https://www.ibm.com/think/topics/artificial-intelligence`, last accessed February 26th, 2025

2.3.3 Summary

This section has introduced the important AI models used in this thesis, namely the Neural Network and one of its most effective subtypes, the transformer. While these types of AI model have driven many recent improvements in performing common NLP tasks, deep neural networks stop being readily understandable to humans rather quickly once the number of neurons and layers is increased. We are interested in how the inputs (words) influence the outputs of AI models when performing NLP tasks. This kind of analysis is the domain of **Explainable AI**, which is why the next section briefly introduces this last necessary field for the approach put forward in this work.

2.4 Explainable AI

Explainable AI (abbreviated: XAI) is the fields of research focused on making the decision-making progress of AI models more transparent and understandable to humans, to enable us to reason about the AI model's decisions [30]. This can be useful to check if the decision process contains social biases, or if it is based on wrong patterns learned from skewed training and test data (overfitting). By analyzing the decision process of high-performing AI models, this work attempts to extract useful information about how the language is processed by them which may be useful to humans as well. For this analysis, XAI is used as a tool to find out which inputs influence the performance of AI models the most.

2.4.1 Distinctions between XAI Methods

There are several categories of XAI methods that will be relevant in this work, because they determine not only the purpose of the method, but also which XAI method can be used on which AI model. This section will go into some of the distinctions and argue why some types are more appropriate to the aim of this thesis than others.

As was mentioned before, this work is interested in analyzing the influence of words in AI model input on its output for the purpose of finding out which words might be the most useful for human learners as well. The question of which inputs are the most important for a model to reach its output is the domain of **feature importance explanations**, a sub-field of XAI concerned with the question: *"how does each feature affect the model?"* [2]. Thus, all XAI methods in this work belong to this group.

Feature importance explanations can be further subdivided into two groups: **Feature selection** methods explain the model's decisions by trying to predict a subset of all input features which are deemed "important". **Feature attribution** methods attribute a concrete importance score $a_i \in \mathbb{R}$ to each input feature x_i. This work uses feature attribution methods exclusively, as the scores will allow us to rank words in a list by their importance.

Another important characteristics of an XAI method is whether it is **model-agnostic** or **model-specific**. A model-agnostic XAI method can be used no matter

[2] `https://courses.cs.washington.edu/courses/csep590b/22sp/files/lectures/lecture2_part1.pdf`, last accessed on February 27, 2025.

what AI model is used, whereas a model-specific method is limited to being used to a specific type of AI model, such as transformers, decision trees [Missing citation about: decision tree], or neural networks. Model-agnostic models are also called black-box-approaches, as they do not look into the model (such as weights in neural networks) but only perturb the inputs and observe changes in the model output. An advantage of model agnostic explanations is that they can be used on any AI model However, many recent state-of-the-art AI models are built using the transformer architecture, hence methods specific to transformer models can also be fairly broadly applied. A major upshot of model-specific explanations is their efficiency: Model-agnostic approaches require us to perturb the input to observe differences in the output, and thus to run the model multiple times for a single explanation, which can constitute a great computational effort. On the other hand, model-specific methods typically only require one run of the model, as they can look into the model itself for an explanation. For these reasons, both model-agnostic and model-specific approaches will be used in this work.

Another feature of XAI methods is the scope of its explanations: **Local** XAI methods explain why a model output was generated for one particular input. A **global** method attempts to reason about the model's behavior in general, independent of any particular input data. This work is interested in observing the interaction of the AI model with a particular corpus, not only the AI model. For this reason, all methods used in this work will be local methods. However, while local explanations only explain a single decision of the model (which, in the work, will be the interaction of the model with a single sentence), we will aggregate the individual decisions across the corpus to get a bigger picture about this interaction.

The transformer attention mechanism introduced in Section 2.3.2 has been used as a model-specific XAI method [15] [14] , and will be used in this work as one among several XAI methods employed to gauge the impact of words in the input to the output of (transformer) AI models. The use of attention as explanation, too, will be discussed in more detail when describing the implementation of our approach in Section 4.4.

Table 2.1 summarizes the use of XAI in this work.

Attribute	**Type Selected for this work**	**Reason for selection**
Model-Specificity (Model-Agnostic or Model-Specific)	Both Model-Agnostic and Model-Specific	Model-Agnostic methods: Broad applicability; Model-specific methods: Computational efficiency
Importance Explanation (Feature attribution or Feature selection)	Feature Attribution	Use of scores to order words in vocabulary lists
Scope (Local or Global)	Local	Attain explanations taking into account the corpus

Table 2.1: XAI methods used in this work

2.5 Ranking Vocabulary Lists

As described in Section 2.1.1, the Routledge Handbook of Second Language Acquisition cites the concepts of frequency, usefulness and difficulty as widely accepted criteria to decide when to teach which words. To the best of our knowledge, all previous methods for making lists of vocabulary for language learners use similar criteria. They can be classified roughly into two categories based on which of the previously named criteria they employ: Approaches which use frequency have the benefit of being easy to implement on a computer and putting a simple number on each word.

Previous approaches which have aimed at generating lists by a utility concept outside of frequency have relied on human experts. This chapter will introduce concrete methods from both categories, and examine each method's strong and weak points. It is this work's contention that AI models make it possible to mechanistically calculate utility in addition to frequency, and that this could be a possible point of attack for refining previously accepted approaches for vocabulary list compilation.

2.5.1 Expert Judgement: Reading Rockets Tier System

This chapter will introduce a traditional approach to vocabulary list generation. While it does put forth criteria for vocabulary selection, they are not considered possible to calculate mechanistically, and thus it relies on the judgement of individual language experts (teachers) to find useful words for their students.

It is seen in the recommendations for vocabulary selection by the initiative Reading Rockets: Reading Rockets [3] is an American national public media literacy initiative by WETA Washington, D.C. [4], a television station of the non-commercial public broadcaster Public Broadcasting Service [5].

As such, one of its aims is to help find reading materials appropriate for teaching children of school age words they are not yet familiar with but which are useful and important, an aim similar to that of finding useful vocabulary for second language learners. Reading Rockets has made public several recommendations for how to find useful vocabulary to teach children as well, with the intention of helping individual teachers select vocabulary for their students. For this purpose, Reading Rockets uses a tiered system to sort vocabulary items into three tiers: [6] Tier One items are words which most children know before even entering primary school and thus need not be taught. This includes words such as *happy* and *baby*.

Tier Three items are words which are useful mostly in one specific domain, such as *peninsula* or *lathe*. These are not high frequency words, and thus Reading Rockets conclude that these words are best learned not in isolation, but rather "when needed in a content area".

Tier Two words are fairly high frequency words that are not so common as to be known by all children, but still useful across many domains, such as *coincidence*, *absurd*. For this reason, Reading Rockets deems that "instruction in these words

[3] `https://www.readingrockets.org/`

[4] `https://weta.org/`

[5] `https://www.pbs.org/`

[6] `https://www.readingrockets.org/topics/vocabulary/articles/choosing-words-teach`, last accessed on March 15, 2025.

can add productively to an individual's language ability." [6]

This system of triage can be useful and could be made to fit with the aim of teaching second-language learners too, by teaching Tier One words first, then moving on to Tier Two and Tier Three words. Because the system proposed by Reading Rockets is a manual one, it requires some effort on the side of a teacher who would like to apply the system to their classroom.

In the next chapter, we will discuss various vocabulary selection methods that utilize analysis of existing textual data to arrive at vocabulary lists.

2.5.2 Frequency-Based Methods

To the best of our knowledge, current algorithmic approaches for vocabulary list generation all rely on frequency as their main criterion for list ordering. However, the raw number of occurrences is not the only metric which falls into this category: The literature offers at least three frequency metrics for vocabulary list creation, namely *Raw Frequency*, *Relative Frequency* and *Average Reduced Frequency*. This section will therefore introduce and discuss these approaches and their advantages and disadvantages. Furthermore, we will discuss *TF-IDF*: A metric which, to our knowledge, have not been used to create vocabulary lists, but for a similar purpose, such as finding words in a document that most characterize its contents.

These computational methods generate vocabulary lists by using corpora and computer-aided language processing (mostly tokenization) to compile vocabulary lists. Compared to the expert-based approach explained in the previous section, they are closer related to the methods that this work strives for, and will be used as points of comparison when evaluating our own approaches.

Raw Frequency

The raw frequency of a word is simply the number of times it occurs in a given text. The idea of making lists of vocabulary by ordering words by their number of occurrences in a corpus is not new: In 1953, English language teacher Michael West published his "General Service List of English Words" [18], and the frequency of the words in a corpus was one of the criteria used to determine its members and ordering.

Nation and Waring are prominent proponents of frequency-based approaches: In their 1997 paper "Vocabulary size, text coverage and word lists" [19], they affirm that the frequency with which a word appears in the target language should be used a metric for its importance to the learner, or utility, and thus teaching high-frequency words should be the focus when teaching beginners, claiming that: "Frequency information provides a rational basis for making sure that learners get the best return for their vocabulary learning effort."

Herein we can see the appeal of frequency as a basis for vocabulary selection: It gives the reader the highest *coverage* of texts, i.e., the percentage of words in the texts they read. However, focusing on maximum-frequency words to achieve text coverage does not take into account that the most frequent words in the language are typically words with little meaning by themselves: The following words are the most frequent ten words on Wikipedia according to a count by Leipzig University [7]:

[7] See "Wikipedia" corpus 2016, `https://wortschatz.uni-leipzig.de/en/download/English`

"the", "of", "and", "in", "to", "a", "is", "was", "the", "for".

Herein, we can see can issue with using frequency as the exclusive criterion for vocabulary selection: We do not contend that the above words are useful and among the most important words to know for an English learner. However, if we were to teach these as the first words, the learner would not be able to understand the meaning any sentence, nor make a complete sentence of their own design. This is because they are prepositions, articles, and copula verbs, which by themselves do not allow for the construction of a grammatically correct sentences, nor convey much meaning. Such words are called *stopwords* in the realm of Natural Language Processing, and they are often filtered out from the inputs for AI models since they are not necessary to perform many NLP tasks [Insert citation: "stopwords being useless"]. Thus, knowing only these words is not sufficient for comprehending most texts.

Possible Improvement: Frequency with stopwords filtered out Given the above issue, we can immediately of a way to remedy the issue: There exist stopword lists for many languages, such as the one provided by the NLTK library [8]. Therefore, we could make a more useful vocabulary list by first counting the frequency of words, and then deprioritizing stopwords by a constant factor, or dispersing them more evenly throughout the vocabulary list, such that the first words already contain nouns and verbs allowing for the construction of (though very limited) grammatically correct sentences.

Average Reduced Frequency

The raw frequency count can give indications about which word in a long corpus are important to know. However, consider two words that both appear in a corpus n times, but one of them only appears in one small section, while the other is distributed evenly across the corpus. In such a case, we might draw the conclusion that the former is less generally useful than the latter.

This is the thought behind **Average Reduced Frequency** (ARF) [26]: If a word occurs n times throughout the corpus, and these occurrences are evenly distributed throughout the corpus, its ARF will be roughly n. However, the more localized its occurrences are, the more its ARF approaches 1. The exact formula can be seen in equation 2.1.

$$ARF = \frac{1}{v} \sum_{i=1}^{f} \min\{d_i, v\} \tag{2.1}$$

where:

f is the raw frequency of the word,

d_i is the distance between each occurrence of the word (included the cyclic interval between the first and last occurence in the corpus), and

v is the average number of tokens between occurrences.

Average Reduced Frequency has been used for the purpose of compiling generally useful vocabulary lists: The KELLY project was a project by multiple European universities aimed at "developing language learning word cards with a language's

[8] `https://www.nltk.org/`

most frequent words corresponding to the Common European Framework (CEFR)" [9]. KELLY employed ARF along with the raw frequency and relative frequency of a word in corpora to find useful words and thus generate lists of vocabulary [13].

TF-IDF

The previous sections have introduced proxy metrics for word utility based on often a word appears in a (single) large corpus. To the best of our knowledge, all current algorithmic methods used for vocabulary list compilation are of this type. The issue with these approaches is that they tend to rate stop words the highest, as these are very high frequency words in any language. Because we are concerned with context-specific language learning, we could attempt to normalize the frequency of a word in a context-specific corpus with its frequency in a more generic background corpus. This is the idea of the **TF-IDF** (**T**erm **F**requency - **I**nverse **D**ocument **F**requency) [21] metric:

It is calculated on a document level: Given a collection of documents, we calculate the TF-IDF of a word by taking the number of occurrences of the word in a document (Term Frequency), but discounting words that appear in a large number of documents overall (multiplying by the Inverse Document Frequency). There are various formulas for calculating the TF-IDF, but for this work, we will use the formula in equation 2.2, as it is a standard version where the discounting of a word is independent of the length of the other documents in the collection.

Given a collection containing N documents, a term t, and a document t, we define $\mathrm{tf}_{t,d}$ as the number of occurrences of term t in document d, and df_t as the number of documents in which term t appears. Then our TF-IDF is defined as:

$$\text{tf-idf}_{t,d} = \mathrm{tf}_{t,d} \times \log \frac{N}{\mathrm{df}_{t,d}} \tag{2.2}$$

(formula taken from [21])

The TF-IDF metric is usually employed for finding keywords in a document, i.e., a small number of words that characterize its contents [21].

The calculated "keyness" can be seen as a proxy for how important a word is for the overall meaning of the document .

We can see that there is a generally frequent words achieve a high rank for the raw frequency metric, but a low rank for TF-IDF. Because of this somewhat contradictory nature of the metrics, we are left to wonder if either of them are good proxy metrics for word utility.

2.5.3 Summary

Current methods do not exploit recent developments in AI technology and thus suffer from several shortcomings: In general, all of them essentially only count words without taking into account their relationship between each other:

Frequency: The most frequent words in texts are often words that carry little meaning by themselves, such as "a", "the", "of" in English. While these may appear in many texts, they are not useful in determining their meaning. TF-IDF: This metric has been used successfully to find the words that give the best hints at a

[9] `https://spraakbanken.gu.se/en/projects/kelly`

text's topic. However, it does not take into account and semantic relationships between the words in a text. Thus, learning words by aggregating TF-IDFs on multiple texts may aid in identifying the topic of texts, but not at finding out what the message conveyed about the topic is. "not" is a highly frequent word in English and thus will have a low TF-IDF score in most documents. But it is essential to know, as it can completely invert the meaning of a sentence.

Chapter 3

Approach

This chapter describes in detail our approach for word utility evaluation to fulfill the goal stated in Section 2.1.2. The problem is first put formally by putting the terms of word utility and vocabulary list efficiency into mathematical objects in Section 3.1. We then describe a hypothetical experiment that could be performed to evaluate word utility using human feedback in Section 3.2. After pointing out the unfeasibility of such a testing method, we suggest ways to perform the experiment with AI models instead of humans as the test subject in Section 3.3, using the technological building blocks referred to previously in Chapter 2.

With a method for utility evaluation established, we then point out why it is in itself not yet sufficient for actually generating useful lists of vocabulary that a human learner could utilize in Section 3.4. To solve this final problem, 3.5 puts forward a general approach for list generation.

3.1 Formal Problem Statement

To repeat the problem statement from Section 2.1.2: The aim of this work is to create computational approaches to find words that have the maximum *utility* given a particular *language context* by means of *proxy tasks*.

Keeping in mind that this is done in order to sort words into vocabulary lists that can be used by language learners, we can conclude that the output of a proposed solution to this problem would be an ordered list of words. If this word list is ordered by descending word utility, we might speak of a *maximally efficient* vocabulary list. We could also put a number on *any* list of words which measures how efficient a vocabulary list is to gain understanding in a given linguistic context. A list should be evaluated as more efficient if useful words appear at the top of the list, and less efficient if less useful words are at the top, since such an order implies that someone learning the words in that order would not gain understanding at the highest possible rate.

We can subdivide the overarching goal of word utility evaluation into three related, but not quite equivalent sub-goals:

- Evaluating the utility of a single word.
- Evaluating the efficiency of a list of words to learn.
- Generating maximally efficient lists of words to learn.

This section will put utility and efficiency into mathematical terms, to provide a clearer explanation of the variables involved in these tasks and their relationship.

Proxy Task

Since utility is defined in terms of language ability, we first need a way to put a number on the language ability of a test subject. This is done by means of a proxy task: We can imagine a human test subject taking a language exam as a proxy task to find out their language ability:

$$\begin{aligned} t : \text{Human} &\to [0, 1] \\ s &\mapsto p \end{aligned}$$

where s is the test subject and t is the proxy task, with a possible performance score p between zero and one.

We can imagine many variables going into this function, such as the time when the test is taken (hopefully the human's performance would increase over time). However, this thesis addresses vocabulary learning, and so the vocabulary of the test subject is provided as an additional parameter to the function t to make a slightly modified function, t':

Proxy Task with Vocabulary

$$\begin{aligned} t' : \text{Human}, 2^W &\to [0, 1] \\ (s, V) &\mapsto p \end{aligned}$$

where W is the set of all words in the language, and $V \in W$ is the vocabulary of the test subject.

Vocabulary List Efficiency

We now have a function t' from the vocabulary to the score in the proxy task (the proxy metric for language ability). Using this function, we can define a measure for the efficiency of an ordered list of vocabulary l containing elements from W.

A list of vocabulary is simply an ordered list whose elements are words of the target language, with no duplicates:

$$l := (l_1, l_2, \ldots, l_n) \quad \text{where } l_i \in W \text{ and } \forall i, j : i \neq j \implies l_i \neq l_j.$$

Note that by this definition, the list does not have to contain all words of the language.

If the test subject learns the word exclusively in order of the list l (and retains them perfectly), their vocabulary at any point will be that list up until some index k:

$$V_{l,k} := \{l_i \mid i \leq k\}$$

This thesis aims at finding vocabulary lists that improve the ability of a language learner at the fastest possible rate. We can call this property the efficiency of the list. To measure it, we can take a number k of words that a learner memorizes from the top of the list. An efficient list means that with a small number of words, the learner should achieve a high score on the language exam (see Figure 3.1). Therefore, we can define the k-word-efficiency of a list as the score that it allows the learner to achieve by learning its first k words.

$$e_{t,s,k}(l) = t'(s, V_{l,k}) \tag{3.1}$$

And with this definition of efficiency, we can define the condition for an optimal vocabulary list given a particular number of words:

$$l_{opt,t'}(s,k) = \arg\max_{l} \quad e_{t',s,k}(l) \tag{3.2}$$

Finding vocabulary lists that approximate an optimal list according to formula 3.2 is the aim of the rest of this work.

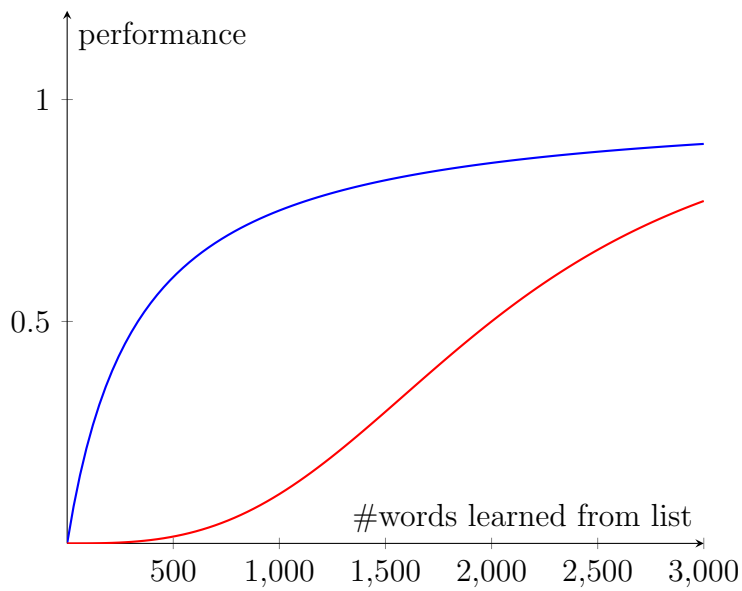

Figure 3.1: Visualization of vocabulary list efficiency: **Blue** represents the score with learning with an efficient list, **Red** represents the score learning from a less efficient list (i.e, less useful words at the top of the list)

3.2 Experimental Setup: Measuring Word Utility as Ability Improvement in Humans

Chapter content: State utility in terms of language ability improvement aim is not just to evaluate, but also efficiently find lists of vocab give problems that is faced when trying to test this approach with human test subjects: no repeatability, high costs

With the formula 3.2, we now have set our goal more concretely. We are left with the question of how to design an actual experiment to test vocabulary list efficiency, and also how to generate one that is approximates an optimal list. Hence, this

section lays forth a hypothetical experiment design by using a human test subject to test efficiency, which will, however, be found to suffer from problems of feasibility, which the following chapters will set out to solve.

Experimental Setup

Let us first consider an example scenario, where we wish to measure the efficiency of one vocabulary list l_A at $k = 200$, $k = 500$ and $k = 2000$ words: First, we would need to find a human test subject s who does not know any words of the target language at the outset. We then make the subject learn the words from l_A one by one. Once the subject seems to know 200 words, we perform a language exam and note their score. Finally, we repeat the process for 500 and 2000 words.

Problems of the Setup

While this test setup might take a long time to complete for high k values, it is still feasible. But if we wish to compare the efficiency of l_A with a second list l_B, we must measure how the same subject performs if they learn from the second vocabulary list, which presents is an unavoidable issue: **The test subject cannot arbitrarily forget the words they have learned from the first list l_A.** The experiment is thus not repeatable.

If we tweak the experiment such that each vocabulary list is learned by a different test subject, we introduce differences in capabilities between test subjects, making the evaluation of the list's efficiencies much more complicated to measure. To make up for these inconsistencies, we might boost the number of test subjects to the point where each list is learned by a statistically significant group of subjects, but this would cause a sharp increase in the cost of the experiment.

However, this theoretical test setup could be feasible using a non-human test subject: Using AI models and Explainable AI to analyze their interaction with language, we could not only evaluate, but even compile lists with drastically reduced costs. This approach is laid out in the following chapters.

3.3 Experimental Setup with AI Model

Chapter content: I view AI models as containers of language knowledge. We can perform studies on it as though it were human, gaining knowledge "from the viewpoint of an entity interacting with language"

This work is interested in finding vocabulary lists that language learners can use to gain linguistic competence in their chosen field in the least possible time. Section 3.1 has specified how with the help of a proxy task, we can define this efficiency. Section 3.2 has suggested a hypothetical experiment for testing the efficiency of a vocabulary list with human test subject. However, we have established that the experiment is not easy to set up consistently, as it cannot be repeated with the same test subject on different lists, and a large pool of test subjects either means lower comparability of the scores or significantly increased costs.

To circumvent this issue, this work proposes the following idea: To replace the human test subject in the experiment described in Section 3.2 with an AI model, facilitating a consistent and economical setup.

The following paragraphs will lay out how, in such a setup, we can model the concepts from Section 2.1.2 such as linguistic context and utility with technological tools.

AI as a Test Subject

In recent years, AI models such as ChatGPT [Insert citation: "cite ChatGPT"] or BERT [Insert citation: "cite BERT"] have become highly adept at fulfilling language-related tasks such as Language Modeling [Insert citation: "LLMs"], Named Entity Recognition and Sentiment Detection.

Such models possess as level of "understanding" that extends to the meaning of words: They can recognize that two words such as "building" and "apartment" are close to each other in meaning, even though the words are dissimilar on a character level (this work makes no claims as to whether this understanding is real to that of a human, only that the models behaves as though it were). Thanks to this semantic language "understanding", AI models surpass the purely statistical approaches that were the paradigm of NLP until recently [Insert citation: "paper contrasting traditional with AI approaches"]. Thus, instead of a human performing a language exam to increases in linguistic understanding (*utility*), we could let an AI model perform an NLP task on a corpus instead.

If we run the model on a specific corpus and mask some of the words in the input, it is expected that the performance will decrease in comparison to the full input. However, presumably some words will have a larger impact on the performance than others. If we view removing words from the input as the equivalent of a human not knowing a word in a text, this performance differential can be a proxy metric for word utility and for human language ability.

Modified Experimental Setup with AI

With this new proxy metric for word utility in mind, we can propose a new experiment. To test the efficiency of a vocabulary list, we modify the setup from 3.2 as follows:

Let a pre-trained AI model perform its NLP task on a corpus. At the start, we mask all the tokens in the input to simulate a language learner who is a complete beginner and thus knows no words in their target language. We then progressively unmask the words from the vocabulary list in the input, and each time run the AI on the corpus to check the updated (and presumably improved) performance.

This will yield a plot of performance over unmasked words which will tell us how quickly the performance improves with unmasked words, similar to Figure 3.1. A quickly increasing performance will correspond to an efficient vocabulary list, whereas an inefficient list will create a plot with low initial gains in performance.

The modified experiment addresses the two issues proposed with the original setup: It is cheap when compared to using a human test subject and unlike the human, the AI model has no memory of previous tests, meaning we can run the experiment consistently every time.

Next, let us see how the components in this setup interact with each other to model the concepts introduces in Section 2.1.2:

Influence of Components on Result

The evaluation of the vocabulary list efficiency in the experiment will depend on the following components:

- The AI model employed
- The input corpus
- The NLP task used

We have already addressed the role of the AI model in Section 3.3. The corpus and the NLP task performed by the model will also influence how quickly the AI model's performance increases with a given vocabulary list:

For example, when performing Sentiment Detection, words relating to emotion such as "hate", "like", "amazing" will matter more than if for an Information Retrieval task such as Temporal Tagging.

The corpus will have a more obvious influence on the performance, since corpora differ in many aspects such as formality, topic, and format of the text, all of which are components of a linguistic context.

Thus, by using diverse corpora, we can model various language contexts, and test word utility in those contexts. Once we have efficient vocabulary lists across those contexts, a language learners could choose one or several of these lists, based on what linguistic context is closest to their language learning goals.

Table 3.1 summarizes the correspondences between the concept and the technical component.

Abstract concept	**Technical Implementation**
Test subject	AI model
Language ability	Performance in NLP task
Language context	Corpus
Word utility	Impact of word on task performance

Table 3.1: Correspondence of abstract concepts to parts of implementation.

The next chapters will tackle the problem of how to not only test list efficiency, but make lists that approach an optimally efficient list.

3.4 Evaluating vs. Making Lists of Useful Vocabulary

The previous chapter describes an approach for evaluating the efficiency of vocabulary lists, which measures how quickly the list lets a learner improve their understanding in a specific context. While it can be seen as an improvement on the human setup because of its improved consistency and economy, it describes only the process of evaluation, not generation. Actually finding optimal vocabulary lists is theoretically possible by testing every possible vocabulary list with all words in the language.

In practice, however, this would require an enormous amount of computational resources, as can be seen on a simple example:

Let us suppose we wish to find the most efficient vocabulary list of 1,000 words for some context. To speed up list generation, we can make a preselection by allowing only the most frequent 20,000 words in the context as candidate members of this list. This would yield a total number of $P(20000, 1000) = \frac{20000!}{19000!}$ possible vocabulary lists, all of which would need to be tested for efficiency to find the most efficient one.

We can optimize the evaluation process, however, since these tests consist of running the proxy task with a gradually increasing vocabulary, and each vocabulary will be used in the evaluation for many vocabulary lists. For our example, a vocabulary is a set of words formed by taking the first n elements of the vocabulary list where $n \in [0, 1000]$. If we only consider the number of possible vocabularies with a cardinality between 0 and 1,000, we get $\sum_{k=0}^{1000} \binom{20000}{k} \approx \frac{1}{2} \times 2^{20000} = 2^{19999}$ runs of the proxy task. Clearly, this is still an unfeasible amount of computation.

It can thus be seen that finding optimal vocabulary list is much too expensive when performed with a brute-force approach. For this reason, the next chapter will introduce an alternative method for finding efficient lists of vocabulary, based on XAI as a tool which extracts information about the linguistic skills of AI models.

3.5 Generating Efficient Lists of Vocabulary

The previous chapters have shown the challenges in evaluating the efficiency of vocabulary lists with human test subjects, and suggested a repeatable approach utilizing AI models, NLP tasks and corpora to overcome these challenges. We have seen, however, that this approach does not suffice to generate efficient vocabulary. Therefore, to achieve cost-efficient list generation, this chapter advocates for adding a final component to our experimental setup, namely Explainable AI.

Explainable AI as a Tool of Analysis

The guiding question of this thesis is: *Which words provide the most language understanding in a given linguistic context?*. Section 3.3 has put forward an experimental setup in which the concepts of language understanding and context are simulated with technological components. By substituting these terms, the question is turned into: *Which words provide the highest improvements in performance of an AI model performing an NLP task on a given corpus?*.

Put like this, the question is very close to the questions that the field of Explainable AI seeks to answer. A typical question of Explainable AI would be: *Why, for a given input, does the model arrive at its output?*. [Insert citation: "Definition of XAI. How to not make this seem like a direct quote?"] To be more precise, this is a question for a *local* explanation (see Section 4.4). As mentioned in Section 4.4, we will use Feature Attribution methods, which for an NLP task transform the question into: *Which words in the input have the most influence on the model arriving at its output?*

By answering this question on a statistically relevant sample of individual inputs from a corpus, we can find out which words are the most essential for the AI model to perform its task in the corpus overall. If the words found in this way are close to

those that would provide the most utility to a human learner as well, they could be used to create very efficient vocabulary lists as well.

Once we have calculated the utilities of words, we only need to sort the words by their utility to arrive at a vocabulary list that should be close to optimally efficient. It must be noted that this is only an approximation of a maximally efficient list. This is because, theoretically, the utility of two words combined could be higher than the sum of their individual utilities. In other words, our approach ignores possible synergistic effects between words.

By adding XAI as a tool for analyzing the interaction of the AI model with the corpus, we finally have an experimental setup for generating efficient vocabulary lists. This is not a replacement of the approach described in Section 3.3. Rather, this work will use the generation approach to make lists (Chapter 4), and the evaluation approach for testing which method leads to the most efficient lists (Chapter 5).

In the next chapter, we will tackle the implementation of this approach. We will present various choices for these components, and argue for some to be used over others, with the aim of making the utility estimated by the framework align as much as possible with utility to human learners.

Chapter 4

Implementation

Chapter 3 has formalized the problem and put forward a novel framework for finding useful words, consisting of two major functionalities: Vocabulary list generation, and vocabulary list evaluation. The list evaluation approach, described in Section 3.3, utilizes the performance of an AI model, corpus, proxy task as a proxy metric for language ability, in order to estimate how efficiently the vocabulary list may help a human language learner acquire competency.

The list generation approach, proposed in Section 3.5, additionally uses Explainable AI as a tool for analyzing the interaction of the AI model with the corpus to compile vocabulary lists that approach maximal efficiency.

In this chapter, we will describe our implementation, mostly of the list generation approach. This is because the list evaluation method will be used in Chapter 5 as one among several metrics for evaluating the efficiency of vocabulary lists generated by the implementation in this chapter. However, the evaluation with our approach will use the same components as the list generation approach, such as the chosen corpora, AI models, and NLP tasks.

The first chapter described the implementation of the list generation approach from a system design perspective, describing the interaction between the individual components. The choice of components in the system is then argued in the following chapters, each of which will feature section describing the desiderata of the component choice are and why, followed by the selection of concrete components. ToDo: Write rest of introduction when chapter structure stands.

4.1 Data Pipeline

This chapter will give a top-level overview of our implementation for the list generation approach described in Chapter 5.

As mentioned before, the main components of this approach are:

- An NLP task, which models a language ability of a language learner.
- An AI model, used as a test subject.
- A context-specific corpus, to model a language context.
- An XAI method, which analyzes the interaction of the AI model with the corpus to determine word utilities.

- If the XAI method allows: A tokenizer, determining the words to be analyzed.

Thus, the implementation of the approach will consist of determining how exactly these components interact, as well as deciding which model, which XAI method, etc., will be used.

The interaction of these components can be seen in pseudocode in Algorithm 1.

Algorithm 1 Efficient List Generation.

Require: corpus, model, xai

```
Initialize line_word_utilities_scores ← empty list
for each line in corpus do
    word_utilities_for_this_line ← xai(model, line)
    Append word_utilities_for_this_line to line_word_utilities_scores
end for
corpus_word_utilities_scores ← aggregate_word_utilities(line_word_utilities_scores)
voc_list ← words ordered by corpus_word_utilities_scores
return voc_list
```

The choice of individual components will be tackled by the following chapters. Because of some dependencies between the components, not all of these components have their own chapter: The NLP task performed depends on the AI model that we use, as most AI models are trained on only one task. Therefore, the choice of AI model is discussed together with the NLP task in the next chapter.

4.2 NLP Tasks and AI Models

The NLP task and the corpus used for list generation

There are many NLP tasks we could choose from [Insert citation: "huggingface NLP task list or something"]. But not all are equally suitable. This chapter will first put forward criteria for selecting NLP tasks in Section 4.2.1 for word utility evaluation. After this, we use these criteria to select a few NLP tasks as appropriate in Section 4.2.2. The NLP tasks we choose will dictate the format of our input data, i.e., the corpora we can use. Therefore, the selection of corpora will be discussed in the next chapter.

4.2.1 Desiderata

Chapter content:

- Support many languages
- Be as close to human need as possible
- → task should show general language understanding
- data is easy to acquire
- → data should be freely available for lots of contexts
- → no need for manual labeling data

This section will put forward four criteria for selecting NLP tasks for word utility extraction: The availability of many multilingual AI models and corpora usable as input for the task, generality of the task and ease of evaluation. These criteria will be used in the next chapter to select concrete tasks for our implementation.

Model Availability in Many Languages

The goal of this work is to find approaches to find useful words for the purpose of language learning. Much research in Natural Language Processing is dedicated to improving NLP performance in English and other high-resource languages such as Spanish, French or Mandarin Chinese [Insert citation: "resource availability in various languages"]. This has the consequence that many AI models and other NLP methods achieve high levels of performance only in these languages, and many AI models are only available in English or only a small number of languages. However, there are over 7,000 languages in the world [Insert citation: "Ethnologue"], and for many of these there exist corpora, or online digital texts which could hypothetically be used as inputs for our word utility extraction approach. For this reason, our implementation strives to realize word utility extraction in **as many languages as possible**.

Corpus Availability in Many Languages

The second point of consideration for task selection is **how much data is available for performing the task**. Ideally, we would like tasks for which suitable corpora are freely available or can be trivially generated from available corpora. This is for the reason that, with a larger amount of usable data, we not only improve the accuracy of our approach, but also increase the diversity of input data. With diverse input data available, we have a larger amount of linguistic contexts for we can find utilities, and the context-specific language learning is a central motivation of this work.

Generality of Skill Required

Another important point to consider when selecting an NLP task is **how general the linguistic skills** are that the task requires: We use the performance in the NLP task as a proxy metric for the test subject's language ability. As such, we must ensure that task reflects a general level of semantic understanding, not only a narrow mechanistic skill that can be accomplished by using only a small part of the input.

Ease of Evaluation

To ensure we can measure the performance, we must also choose a task whose results can be easily compared with each other: Some tasks, such as text summarization, present a challenge for automatic evaluation: It is difficult to put a number on how similar two summaries of a text are. While evaluation measures, such as BLUE scores[Insert citation: "BLEU"], exist, it is questionable how well they capture the similarity between texts, because they do not recognize the semantic similarity of synonyms, and a different sentence structure will result in a low BLEU score even

if the actual meaning of the sentences may be very close. It follows that if we have the freedom to choose NLP tasks whose results can be automatically evaluated with a fair degree of accuracy, we should choose them.

Summary

To summarize our desiderata for NLP tasks: Our implementation seeks to use NLP tasks for which both AI models and corpora exist in a large number of languages, to maximize accuracy and diversity of linguistic contexts which can be modeled. We prefer tasks that demonstrate general language understanding over tasks that only require a narrow skill set to perform or that are too technical, because general tasks are expected to align more with human linguistic skills. Finally, the task must be easily scorable, since the task score is the metric by which we gauge how useful words are.

The next section will present several tasks which fulfill the above criteria, and explain what consequences their selection has on the other components of vocabulary list generation. The choice of NLP tasks employed to test a XAI-based approach for word utility estimation is a crucial step: Since we are trying to estimate utility by measuring the impact of a word on language understanding, the NLP tasks should reflect language understanding as much as possible.

4.2.2 Task Selection

A good place to start looking for such tasks are those which are typically employed for pre-training NLP models: Pre-training tasks are used to first endow the AI model with a general understanding of the language, before using transfer learning to specialize it for a more specific downstream task. Such tasks must necessarily be general and require general language understanding, since training the model with them is supposed to provide a solid basis for a wide variety of NLP tasks. Another benefit of using pre-training tasks is that their training is unsupervised, meaning there is no need to manually label data. ToDo: Look at various pre-training tasks, preferably those used by state-of-the-art AI models ToDo: include free availability for AI models in their justification

4.2.3 NLP Pre-Training Tasks Used by State-of-the-Art AI Models

This section takes a look at the pre-training process of recent state-of-the-art LLM models which have made public their training process. Both the NLP tasks and the kind of data is considered.

GPT-4 [20]

Task: Language modeling (see next section).

Data: Not disclosed in detail, according to the original paper, the model was trained "using both publicly available data (such as internet data) and data licensed from third-party providers".

GPT-3 [4] GPT-3 is a model that does not rely on transfer learning to apply its linguistic understanding to new tasks; instead, it uses zero-shot and few-shot learning to perform tasks it was not specifically trained for.

Task: Language modeling (same as GPT-2 [22])

Data: Common Crawl, WebText2, Books1, Books2, Wikipedia ToDo: link sources?

LLama 3.3 [1]

Task: Meta did not make public the training process for Llama 3.3.

Data: "data from publicly available sources"

4.2.4 Tasks Considered

Next Sentence Prediction In this task, the AI model takes as input two sentences and predicts a probability for the second sentence being the successor of the first sentence in their source text. Advantages for this task for our purposes is that such a dataset is easy to generate, as it merely requires a corpus of sentences that follow from each other, which is easily obtained from Wikipedia articles, film subtitles, or any other continuous text.

Text summarization This task involves summarizing a given text, in other words, writing a shorter version of the input text while still conveying as much of the information from the original text as possible. Summarizing texts seems to require a high level of "understanding" of the text and would thus seem to be good choice for testing whether ablating certain words from the text would have detrimental effect on the model performance. Unfortunately, this task requires hand-labeled datasets and is thus not a good candidate if we aim to find approaches which can be implemented in many different languages, as there is a dearth in data in many of the less-studied languages of the world.

Masked language Modeling (aka. "cloze task")

Causal Manguage Modeling (aka. Next token prediction)

Sentence order prediction

Sentence embeddings Sentence embeddings take the approach of transforming words into meaningful vectors and extend it to whole sentences. This "task" differs from the others in that we do not measure differences in performance when the input is perturbed; but rather a distance between the embedding vectors themselves. This justification for such an approach is that sentences whose meaning is very different should end up further apart from each other in the vector space once embedded. This brings several advantages: This approach can be performed on any corpus containing distinct sentences. These corpus does not have to be document-level, and sentences need not be consecutive. To make this a task on which XAI methods can be applied, we can define a distance from the original token

4.2.5 Sentence Embedding Methods

LASER [3]

BERT [24]

4.2.6 Data Required for Each NLP Task

The various NLP tasks employed require certain types of corpora to be employed properly:

xt sentence prediction Requires a corpus that contains consecutive sentences. Furthermore, NSP typically predicts whether two sentences follow each other in a document, not a dialogue (see the data on BERT training [12]). This excludes movie subtitles from the possible corpora for this task.

4.2.7 AI Models

- NSP-model ABC
- LLAMA?

ToDo: tests of performance tests of models used with corpora used (e.g., if NSP prediction model is reliable)

4.3 Corpora

The previous section has put forth criteria for which NLP tasks should be used in our word utility evaluation approach. The other core component of the approach is the input data used to perform the experiments, as it serves the purpose of modeling the language contexts in which the language learner is striving to achieve proficiency. This chapter will therefore first lay out the general selection criteria for corpora. We will then introduce some publicly available corpora which are suitable for our evaluation approach, as well as propose data augmentation methods to increase their usefulness further.

4.3.1 Desiderata

Chapter content: For the purposes of this work, it was desirable that the corpora used be:

- representative of what language learners strive for
- freely available in many languages
- can be split up into diverse linguistic contexts
- for NSP: must be comprised of documents for continuous sentences

This section will put forward our general criteria for corpus selection, following from our overarching goal of using the model to model linguistic contexts for the purpose of language learning. ToDo: when the following subsection stand, summarize here ToDo: when actual corpora are decided: Mention them in throughout this chapter too

Document-Level Corpus

The first desideratum for our corpora stems from our choice to use Next Sentence Prediction as one of our NLP tasks: As Next Sentence Prediction predicts whether one sentence is likely the continuation of another, it requires continuous sentences pairs to work. To generate these pairs, we must use (at least some) corpora featuring continuous tests, not only individual sentences. This presents a challenge, as many corpora which are compiled from Web Crawls, such as the Common Crawl dataset [1], present independent single lines, mostly because single sentences cannot be put under copyright in many states [Insert citation: "copyright being the reason for non-document level corpora"] One solution to this would be make a dataset from crawling various news pages ourselves. However, Wikipedia contains articles on many different topics which do not underlie strict copyright license (see the section on Wikipedia), thus for this work, we opted not to compile our own document-level corpus.

Free Availability in Many Languages

In Section 4.2.1 we put forward reasons for why freely available AI models which can handle a diverse pool of languages are desirable for our undertaking. By the same token, we also prefer corpora which are publicly available in many languages over corpora which only include data in one language. While a monolingual corpus is not inferior to a multilingual one, its use necessitates that the user manually compile many corpora if a multilingual implementation is to be achieved.

Closeness to Linguistic Contexts Desired by Language Learners

As mentioned before, the corpus in our word utility extraction approach serves the purpose of modeling a linguistic context, and this linguistic context should reflect some set of situations that a language learner is likely to find themselves in. Typical situations would include reading the news, reading literature or watching movies in their target language. As such, corpora which are close to the materials which language learners are likely to engage with are desirable, since their use makes the AI model's performance on the task more reflective of skills that a language learner would like to acquire.

Diversity of Linguistic Contexts within Corpus

Not only the relevance of the entire corpus's linguistic context is important: Some corpora enable us to further split them up into smaller corpora with more specific language contexts. For instance, while we can use a corpus such as *Wikipedia* as a whole, the structure of Wikipedia enables us to group articles by the category they

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

belong to (politics/sports etc.), as well as split them by subheading (History/Introduction/etc.) to find more specific contexts. Such corpora are especially efficient for generating multiple contexts, hence we will make use of such corpora.

4.3.2 Corpora Used

Background Corpus: mOscar

For the implementation of TF-IDF, we require a generic background corpus to normalize raw word frequencies found in context-specific corpora (that is, to calculate the IDF part of TF-IDF). For this purpose, corpora crawled from the internet offer themselves as a diverse option: While we cannot be certain if some kinds of web pages will feature disproportionately in the final corpus, web content is diverse in that it encompassed both formal and informal content on many different topics.

A natural choice of diverse web content is the *Common Crawl* dataset [2]. However, in its raw form, it takes the form of HTML code and includes much boilerplate content such as cookie requests, necessitating much preprocessing. Another major downsize of *Common Crawl* is that its data is not categorized by language, meaning we would have to perform language categorization as an additional preprocessing step before using it for our purposes. We also rejected the *Wortschatz Leipzig* Web Corpora because, while these corpora are stripped of any HTML code, they are lines taken randomly from websites, rather than whole documents. The original paper for the *Wortschatz* corpus also mentions political content was utilized to bootstrap in the web searches for data gathering, including the *Universal Declaration of Human Rights* and the Jehova's Witnesses magazine *Watchtower* [3], which may lead to a respective bias in the final corpus.

Finally, a corpus from the *Oscar* project [4] was chosen as a background corpus: Each *Oscar* corpus is a cleaned-up version of a *Common Crawl* dataset, on which preprocessing steps such as HTML stripping and removal of boilerplate content have already been performed. It consists of (whole) web pages, and most importantly, its content is language-tagged, covering more than 150 languages in total. There exist many versions of the Oscar dataset, but this work, the *mOscar* [6] corpus from 2024 was chosen, both for its recency and coverage of many languages.

ToDo: State number of documents used from Oscar corpus

OpenSubtitles Parallel Corpus

This set of corpora contains parallel corpora: Corpora which has text segments in one language aligned with the presumed translation of the segment in a second language. Its sentences are generated from subtitles from the popular subtitle sharing platform *OpenSubtitles* (https://www.opensubtitles.org/) and undergo various preprocessing and filtering steps as described in [17]. These include:

1. Enforcing universal UTF-8 character encoding.

2. Splitting and joining of sentences from their original subtitles blocks (the segments which appear on-screen when watching the movie with its subtitle). One

[2] `https://commoncrawl.org/`
[3] Previously available at `watchtower.org`
[4] `https://oscar-project.github.io/`

such block may contain multiple sentences, or only a partial one. There is thus an n-to-m-relationship between the blocks and sentences.

3. Checking and correcting possible spelling issues, especially ones arising from OCR (Optical character recognition) errors.

4. From available subtitles, identifying the subtitle pair which is most likely to be accurate in its alignments and free from errors such spelling, taking into account metadata such as user ratings of subtitles.

One advantageous aspect of this corpus is that it contains many sentences that are sequential, which means we can generate a Next Sentence Prediction dataset from it (add hedging here since not all lines in corpus are sequential and even within the same movies there may will be pauses in the subs). This corpus has been used to train machine translation models such as OPUS-MT [28], a freely available set of transformer models for translation, including between low-resource languages. ToDo: Not completely correct, the pipeline uses data from OPUS, not necessarily or specifically from OpenSubs While it is possible to reconstruct which movies the subtitle lines came from information contained in the corpus, it is unfortunately not clear how these movies were selected in the first place.

The full process, as illustrated by the authors, can be seen in Figure 4.1 As of 2025, the latest version of the corpus (v2018) contains aligned subtitles of 62 languages between each other.

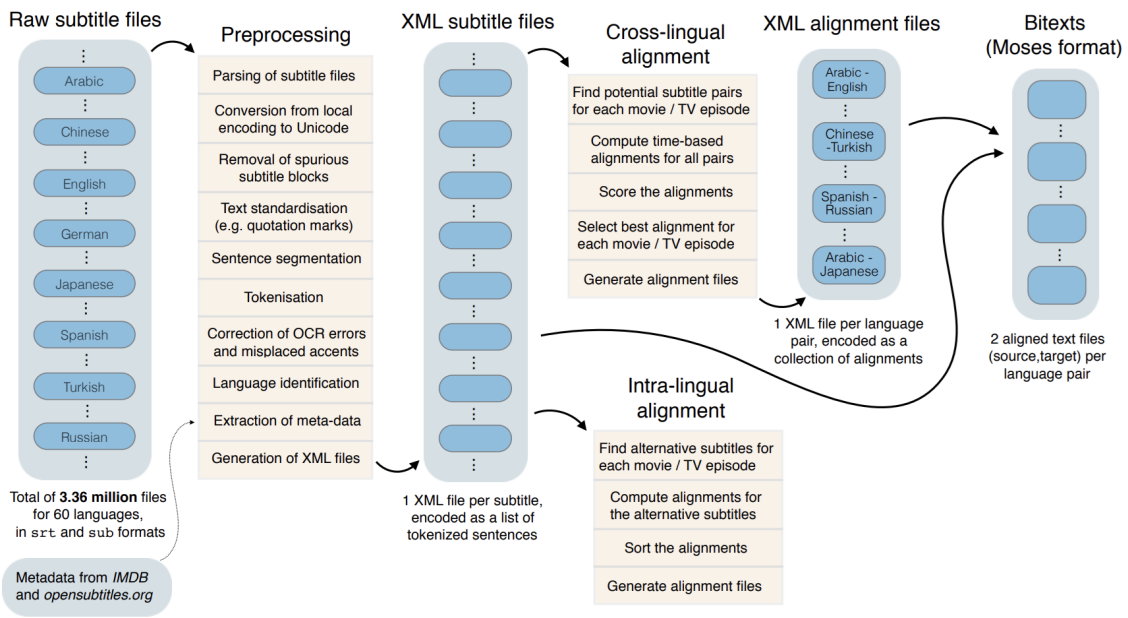

Figure 4.1: The pipeline producing the OpenSubtitles parallel corpus

Wikipedia

Chapter content:

- advantages: diversity of topics, document level corpus which is not copyrighted, regularly updates dumps

- disadvantages: language is technical, does not resemble spoken conversation

- describe grouping intro multiple scope corpora: describe category structure of Wikipedia

Wikipedia offers a large amount of data which can be downloaded in the form of dumps at `https://dumps.wikimedia.org/`. In this work, these dumps will be valuable in modeling various linguistic contexts that a language learner might be interested in.

Advantages: Wikipedia is a valuable resource for several reasons: It contains a vast number of articles, from a diverse set of topics. Wikipedia articles have categories assigned to them, which we can use to group articles into linguistic contexts. This means we do not have to assign the articles to topics ourselves, thus enabling the creation of many smaller-scope corpora from Wikipedia with minimal effort. The characteristics of the article category system on Wikipedia will be discussed in the paragraph below.

Another upshot of Wikipedia is that its articles are licensed under the Creative Commons Attribution-ShareAlike 4.0 International License (*CC BY-SA*). Because of this, Wikipedia can provide dumps of its entire article database ready for download, with the articles in continuous form and with category metadata attached to them. This enables us to use the articles as a source for Next Sentence Prediction NSP task.

Disadvantages: While Wikipedia offers a large diversity of articles and article topics, the language of the articles follows an academic style of writing. Thus, the diversity in registers is comparatively low: Wikipedia is less likely to contain informal language, such as nonstandard language (English: *ain't*, *y'all*) and slang.

Furthermore, it features little language taking conversational form. This is especially relevant for languages where the certain grammatical forms signify something about the relationship between the speaker and listener: In Japanese, there exist various linguistic markers of politeness (*Teineigo* and *Keigo*) which play a crucial role in in-person interactions between Japanese speakers, as their presence or absence can signify respect, familiarity or humility with regard to one's interlocutor. However, such forms are rarely found on Wikipedia, as it does not take the form of a person addressing another person or persons. Thus, we can see that texts on Wikipedia are of limited use to model linguistic contexts involving in-person conversion.

With these advantages and disadvantages in mind, this work uses Wikipedia articles and their associated metadata to make corpora of various sizes, as will be seen in the next section.

Modeling Linguistic Contexts One of the chief advantages of Wikipedia as a corpus its diversity of covered topics. A language learner who is interested in a particular topic will likely find a matching article category on Wikipedia: For example, a person with an interest in American cinema who is learning English could likely take vocabulary from the articles of the category "20th-century American actresses" to boost their language learning in that linguistic context. For this reason, we use a number of manually selected categories as testing grounds for our list generation approach. For each category, we use a number of articles to make a list, then use different articles from the same category as testing data in the evaluation.

We can thus check if learning vocabulary from a category genuinely prepares a learner in understanding texts from that context.

At the same time, a learner could also wish to make a vocabulary list from concrete Wikipedia articles they would like to read. For that reason, also use single articles as corpora to be processed by our list generation approach. In that case, no split of training and test data is necessary, as the aim is not to model general knowledge about a domain, but equip the learner with vocabulary about the **closed** context of a Wikipedia article. Therefore, in the single-article approach, both the list generation and evaluation use all lines of the article.

Leipzig Wortschatz Corpora

Available in x languages

But: data quality issues, methodology might be outdated

[7]

CCMatrix / NLLB

4.4 XAI Methods

In order to appraise the influence of words in a corpus on the performance of the AI model performing and NLP task, this work employs Explainable AI methods, as explained in Section 3.5.

4.4.1 Desiderata

Section 2.4 has explained the use of XAI and stated what kind of XAI methods are especially appropriate for our purpose: We will employ local, feature attribution XAI methods of both model-specific and model-agnostic type, as these will allow use to gauge the importance of words in the input from multiple points of view. Model-specific methods have an advantage in that they typically do not require many model calls to attribute importances to features, and take the model structure into account. On the other hand, model-agnostic methods are useful in that they can be used on any model and gauge the input-output relationship by performing actual experiments on the model, by observing the way changes to input change the output of the model.

In addition to these taxonomical specifications, an ideal XAI method for word extraction should be both accurate in its feature attributions and should not require extensive computation. Apart from the benefit of attaining faster results, a computationally lightweight approach allows the model to process more data in less time, presumably boosting the accuracy of the resulting utility evaluation.

In the following sections, we will propose several concrete XAI methods for use with the described list generation approach. The relationship of the XAI methods to the NLP tasks will also be discussed, as not every XAI method is feasible with every task, either because of unrealistic amount of computation or because the results of the combination are difficult to evaluate. The results of the experiments with these methods will be seen in Chapter 5. ToDo: argue against approaches where the input is reduced to BOW, such as SHAP

4.4.2 Model-Agnostic Approaches

Chapter content: Single Token Ablation, Progressive Summary with

Single Token Ablation

Our aim in using XAI methods is to find out which words, if known, improve the model's performance the most, when compared to not knowing the word. Perhaps the simplest way of testing this is by going through every word in the input to the AI model, and checking the model performance when the word is removed from the input. Within this work, this approach will be known as **Single Token Ablation**.

Approach: Given an input line or a pair of lines: For each word in the input, create a variation of the input where the word is removed. Run the model on all inputs and measure how much performance is degraded in each case. Estimated utility is the difference in performance between not removing and removing the word.

Advantages: One advantage of Single Token Ablation is its conceptual simplicity, as it is very easy to implement. Furthermore, its time complexity is fairly minimal for a feature method, as it requires only one model call for each unique token in the sentence ($\mathcal{O}(n)$).

Disadvantages: The main downside of this method is that, in long sentences, leaving out a single token will not change the meaning of the sentence to such an extent that the sentence becomes unintelligible. Assuming the AI model can process sentences with missing words with similar accuracy as humans, the impact on performance may therefore be minimal or not predictive of actual usefulness. Furthermore, this method does not take into account the utility of a word in the context of word combinations.

Summary: Among model-agnostic methods, Single Token Ablation is a computationally inexpensive whose feature perturbation consists of ablating single words. We may think of this method as simulating a language learner who already knows all words in a sentence except for one, and as such it may not model word utility in such a way that the resulting list is useful for beginners.

Progressive Summary

While Single Token Ablation is easy to implement and cheap to execute, its downside is that the performance differences may not be representative of actual utility. To achieve a more accurate evaluation, we may think of an approach that is in some sense the inverse of removing single words from sentences: Checking which words are the most essential by reducing the sentence a minimal number of words.

Approach: Given an input line, we first let the AI model run on the unaltered input. We then go through every word in the input line, let the model run with the single word as input, and check which of the words produces an output that is close

to the output for the unaltered line. We can thus see which word summarized the sentence the best.

However, we need not stop here: After determining the most essential word in the sentence, we can check which words from the remaining pool of words brings the outputs closer still. By iterating this method, we progressively reconstruct the original sentence, which each step being the best next word to add in order to restore the original output. For this reason, we will refer to this method as **Progressive Summary**. Here, the estimated utility of a word is the boost in performance which is brought to the sentence when the word is added.

Advantages: The Progressive Summary approach models the perception of a beginner-level language learner better than Single Token Ablation. The latter asks which single word, when lacking, degrades performance the most. Progressive Summary instead asks which words, if known, bring language performance closest to knowing all words in the sentence.

Disadvantages: Progressive Summary is much more computationally intensive than Single Token Ablation, which requires only one model call per word in the sentence. Because of its iterative approach, for a sentence with 10 words, Progressive Summary calls the model $10 + 9 + 8 + ...$ or $\sum_{i=1}^{n} i = \frac{n^2+n}{2}$ times, which is a time complexity of $\mathcal{O}(n^2)$.

Compatibility with NLP tasks As Progressive Summary starts from a set of one words and then progresses to larger sets, it is difficulty to meaningfully employ in combination with an NLP task whose input is not a single line, e.g., Next Sentence Prediction. One could circumvent the issue by leaving one of the sentences in its original form and only perturbing the other, but this can hardly be seen as a modeling of natural language ability.

ToDo: Maybe: Describe issues of using PS with models whose output is single number: Approximating with single words may not be meaningful approximation at all. But should test empirically.

As such, our implementation Progressive Summary is not used with Next Sentence Prediction.

Summary: Progressive Summary attempts to find the most important words in a sentence one by one, by first reducing the sentence to one word, then two etc., and using a distance metric to compare the "summaries" to the original sentence. The words which bring the model output for the summary closest to that of the original sentence are regarded as the most useful.

4.4.3 Model-Specific Methods

The model-agnostic methods so far are general tools which can be employed on any model. However, as many of the latest state-of-the-art AI models follow the Neural Network architecture, and more specifically the transformer architecture (see Section 2.3.2), we will also attempt word utility estimation with methods specific to these model. These model-specific XAI methods have the advantage of being computationally efficient: Model-agnostic methods extract information about the

model by perturbing its inputs and analyzing the resulting changes in the output. This requires many model calls for one input datum. In contrast, model-specific methods can analyze the input-output relationship with only one model call, as they look into the model itself for information. We will therefore try two model-specific methods for analyzing word utility as well, as their more efficient calculations means they can process more input data in a given time, which should increase the accuracy of the resulting word list.

Attention as Explanation

One XAI method which has been proposed to explain the decision of transformer models specifically is the transformer's own attention mechanism: As stated in Section 2.3.2, transformer model use attention to focus on some inputs over others. This may be used as a kind of explanation, as the inputs which receive more attention presumably contribute more to the model's decision than those with less attention.

Approach: Let a transformer model run on an input line. The transformer will attribute an attention value to each token in the input. Normalize the attentions such that the attention for all words in the line sums up to one. The attributed attention to a word is its estimated utility.

Advantages: As hinted at before, this method uses one model call for one input line, and thus has a time complexity of $\mathcal{O}(1)$.

Disadvantages: One disadvantage of Attention as our attention mechanism is that attentions are attributed to input tokens of the model. Unlike the black-box methods where word masking is controlled by a tokenizer *independent of the model*, the tokenizer cannot be changed. AI models are trained with some tokenizer as preprocessing the input, and this tokenizer cannot be changed, lest we change the input format to the AI model itself. This is an issue because it means that lists generated with the model-specific tokenizer are not necessarily comparable to lists from model-agnostic methods, as the tokenization method may differ. Additionally, many tokenizers which are used in the preprocessing of models tokenize on a *sub-word* level:

In addition to this technical downside, Attention is controversial as a mechanism to explain model decisions. [Insert citation: "Attention is not Explanation"]

Summary:

Layer-wise Relevance Propagation (not yet implemented → may be deleted)

Approach:

Advantages:

Disadvantages:

Summary:

Chapter 5

Evaluation

5.1 Evaluation Measures

The various utility extraction approaches produce ranked lists as outputs. To compare these, we employ both quantitative and qualitative comparison approaches, described in the following chapters.

5.1.1 NLP Task Performance

The primary metric for evaluating a vocabulary list's efficiency will be the approach explained in Section 3.5: We let an AI model perform an NLP task with a vocabulary consisting of the first k elements of the list. The limited vocabulary is simulated by removing words outside the vocabulary from the inputs to the model. We then progressively increase the vocabulary by increasing index k until we reach the end of the list. The increase of vocabulary size is exponential, as this will ensure a quick evaluation and performance is expected to not grow very much towards the end of the list, where the less useful words should be (this assumption can be confirmed in the evaluation results in Section 5.3).

Listing 2 shows pseudocode for the described setup:

ToDo: Edit surrounding descriptions of efficiency to be in line with pseudocode

This approach gives us a direct, quantitative metric for the efficiency of a vocabulary list.

5.1.2 List Similarities

In order to compare the results provided by the various approaches, metrics are needed that can be consistently calculated across the different approaches. While a human may be able to qualitatively analyze lists and gain a rough idea of their similarity, computed metrics provide an instantaneous (if simplified) outlook on similarities. A metric shall be defined as a function which takes as parameters two word lists of equal length which are word lists ordered descendingly by supposed relevance, and outputs a real number giving either a distance or similarity between the lists.

Considerations of the choice of metric are:

Handling of lists with partial overlap. Metrics must be able to handle elements

Algorithm 2 List Efficiency Evaluation.

Require: voc_list, corpus, model
1: $INIT_VOCABULARY_SIZE \leftarrow 10$
2: $VOCAB_EXPANSION_FACTOR \leftarrow 2$
3: $vocabulary_size \leftarrow VOCABULARY_SIZE_INIT$
4: Initialize $vocabulary_scores \leftarrow$ empty list
5: **while** $vocabulary_size < length(voc_list)$ **do**
6: vocabulary_size $\leftarrow vocabulary_size * VOCAB_EXPANSION_FACTOR$
7: $vocabulary \leftarrow voc_list$ until index $vocabulary_size$
8: Initialize $line_scores \leftarrow$ empty list
9: **for** each $line$ in $corpus$ **do**
10: $line_with_only_known_words \leftarrow line$ without words not in $vocabulary$
11: $baseline_output \leftarrow$ model($line$)
12: $new_output \leftarrow$ model($line_with_only_known_words$)
13: $line_score \leftarrow$ similarity($baseline_output$, new_output)
14: Append $line_score$ to $line_scores$
15: **end for**
16: $vocabulary_score$ = avg ($line_scores$)
17: Append ($vocabulary_size$, avg_score) to $scores$
18: **end while**
19: **return** $scores$

which occur in only one of the two lists. Thus, a metric which solely compares ranks of elements is not viable.

Start of lists is more impactful than end. Since the beginning of lists contains the words which are ranked as most important, changes at the top should impact the metric more than changes at the bottom. This includes (1) Equal differences in rank should be counted as more important if they occur further up the list. A word that is rank 1 in list A but rank 101 in list B says more about the similarity than if a word is rank 2000 in list A but rank 2100 in list B. Likewise, if a word is absent from list B, it implies a greater difference if that word is at rank 1 in list A than if it were at rank 1000.

Metrics Used

Sequential rank agreement (modified) [5]: This metric is based on the deviations of some subset of the lists in the upper ranks. It is important to note that this metric has an additional parameter "depth" which determines how many elements (from the top of the list) are considered. It is therefore more helpful to view its results at various depths. The original formula for this metric in the case of two lists is:

$$a_d := a \text{ from start to rank } d$$

$$S_d := a_d \cup b_d$$

$$SRA_d(a,b) := \lambda \cdot \frac{\sum_{x \in S_d} \sigma^2 \left((r_b(x)) - (r_a(x)) \right)}{|S_d|}$$

where λ is a normalization factor ensuring that $\max(SRA) = 1$. In its proposed form, this metric can only compare lists which contain the same set of unique elements, just in different orders. In order to make it work on lists where this is not the case, one can set the "rank" of nonexisting elements to a value greater than the length of the lists, such as $2|a|$. Another drawback of the metric is that the standard deviation of two numbers does not depend on their absolute value, only their difference. However, to satisfy number 3 of the stated requirements, we can take the deviation of the logarithm of the ranks instead of the deviation of the ranks themselves, resulting in the formula

$$r'(x) := \begin{cases} \text{rank}_b(x) & \text{if } x \in b, \\ 2 \cdot |a| & \text{otherwise.} \end{cases}$$

$$SRA_d^{mod}(a,b) := \lambda \cdot \frac{\sum_{x \in S_d} \sigma^2 \left(\log(r'_b(x)) - \log(r'_a(x)) \right)}{|S_d|}$$

For this modified version, λ can be calculated with:

$$\lambda = \frac{1}{SRA_d(a, a^*)},$$

where a^* is a list such that $a \cap a^* = \emptyset$.

Discounted Cumulative Gain : This formula outputs a value between 0 and 1, with 1 being given if both lists are identical, 0 when they have no elements in common, and values in between when there is partial overlap between elements and/or their order is different. $DCG_p = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i+1)}$

$$rel_i := \begin{cases} \frac{1}{rank_b(el_i)+1} & \text{if } el_i \in b \\ 0 & \text{otherwise} \end{cases}$$

Metrics Rejected

Kendall rank correlation [11]: This metric is bounded between 0 and 1 and compares the ranks of the elements of two lists. However, it cannot handle elements that only occur in one of the two lists, and thus is not suitable for our purposes. It also does not distinguish between differences in the upper and lower parts of the lists.

Spearman's footrule [27]: Rejected for the same reasons as Kendall rank correlation.

Tests of Applicability

[Results of preliminary tests of various metrics on own lists such as rank switching ,replacement at bottom and top of list etc.]

5.2 Baselines

It may be useful to compare the lists generated by the various approaches with existing word lists from educational materials: Textbooks often feature chapters with word lists, or sentences which can be converted to word lists with a tokenizer. The purpose is to have a point of comparison, to see if generated lists agree with existing lists, and find reasons for differences.

5.2.1 Existing Lists

As one point of comparison, we can use existing vocabulary lists that are readily available in digital format, regardless of the method which created them. For this, a survey was done on popular language learning tools which make public some of their learning data.

ToDo: Language learning textbooks

Duolingo

While Duolingo is the most popular language learning application as of 2024, it does not publish its word lists or course contents that is free of cost and easily convertible to a format that can be processed with NLP tools.

Rosetta Stone

Rosetta Stone[1] is a popular online and mobile language learning platform. *Rosetta Stone* publishes some of their course contents on its website [2]. These lessons take the form of phrases that should help a learner to quickly gain competence in their target language. To make a list of vocabulary from *Rosetta Stone*'s Lesson Contents, we use a tokenizer on the lesson and let the order of first appearance of a wordi n the lesson be its rank in the resulting vocabulary list. Thus, the generated list features those words first which *Rosetta Stone* introduces first to its learners, which presumable is close to the word order that *Rosetta Stone*estimates is most useful. One qualification of the previous statement is that some words are probably only introduced so the learner has something to talk about, rather than because of the word's inherent utility.

This chapter has introduces some pre-existing lists as points of comparison for our generated vocabulary lists. However, one shortcoming of this comparison is that these lists have not been prepared with the same linguistic contexts in mind as those made by this work's approach, and thus their utility is naturally greater in general-context corpora than specialized ones. While the context specificity of this work's utility extraction method is a major advance for creating vocabulary lists, it also makes this an unfair comparison.

However, apart from pre-existing lists, there also exist pre-existing **methods** of finding important words in texts, which we can also use as baselines to compare with our method. These also use corpora, but not AI models or XAI methods to find important words. The next chapter will introduce some of these methods, which we will use to generate comparison lists to our own.

[1] `https://rosettastone.co.jp/`

[2] `https://support.rosettastone.com/s/article/Rosetta-Stone-Course-Content`

5.2.2 Existing List Generation Methods

To get an impression of how well our XAI-based method for vocabulary list generation performs, it will be informative to test established methods for extracting important words from texts.

Raw Frequency

Raw word frequency has been used as a criterion for which words to teach to language learners [8]. Frequency is much less computationally intensive than any of the XAI-based approaches, as it only requires a tokenizer to be run on a corpus, without involving model calls.

Frequency, Without Stopwords

One drawback of using raw frequency the criterion by which to order vocabulary lists is that he most frequent words in most corpora will consist of stopwords. This can be seen in the following frequency word list from corpus [x]. ToDo: insert sample from raw frequency word list from corpora used in experiments

These words, while being the most generally useful, have very little meaning by themselves [23]. For this reason, they are often filtered out in the inputs of NLP tasks. Removing from the raw frequency-based word lists could be a way boost the lists's efficiency, especially at the top of the list. We therefore filter out known stop words, based on a list from the popular NLP library *NLTK* [3]. [4]

TF-IDF

ToDo: either finish this section or replace with collection frequency normalization
To characterize the contents of a document, [21]. TF-IDF has, to the best of our knowledge, not been considered for vocabulary list generation. However, it is often employed to find key terms in a document, which characterize the document's contents with a small number of words.

TF-IDF: The frequency of a word in the context-specific corpus, normalized against a background corpus.

5.2.3 LLM Prompts

In recent years, Large Language Models have become a popular tools for language learners to find new words to learn about specific areas. For this purpose, we run the following prompt to ChatGPT 3 for each context to get a sense of how well my method performs against this easy method. ToDo: Prompt and results

5.2.4 Sample text

ToDo: Show a sample paragraph with first n words from each list. Can be used to intuitively evaluate utilities

[3] `https://www.nltk.org/`

[4] The NLTK stopword list can be downloaded by following the instructions on `https://www.nltk.org/howto/corpus.html#word-lists-and-lexicons`.

5.3 Results

Chapter content: Discussion/Analysis [Interpretation of the results] Could you combine approaches with each other or baselines to combine strength, what are the strengths/weakness of each approach

5.4 Discussion

Chapter 6

Outlook

- Sentence importance evaluation
- Capturing text as coherent unit instead of sentences which are independent from each other
- Active vs passive utility: Ideas for active utility estimation
- Compound splitting
- Lemmatization
- n-grams

Chapter 7

Appendix

7.1 Abbreviations

NSP Next sentence prediction

NLP Natural language processing

LLM Large language model

Bibliography

[1] Llama-models/models/llama3_3/MODEL_CARD.md at main · meta-llama/llama-models. https://github.com/meta-llama/llama-models/blob/main/models/llama3_3/MODEL_CARD.md.

[2] Alexander Clark, Chris Fox, and Shalom Lappin. The Handbook of Computational Linguistics and Natural Language Processing: Introduction. In *The Handbook of Computational Linguistics and Natural Language Processing*, pages 1–8. John Wiley & Sons, Ltd, 2010.

[3] Mikel Artetxe and Holger Schwenk. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the association for computational linguistics*, 7:597–610, 2019.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, and Amanda Askell. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[5] Claus Thorn Ekstrøm, Thomas Alexander Gerds, Andreas Kryger Jensen, and Kasper Brink-Jensen. Sequential rank agreement methods for comparison of ranked lists, August 2015.

[6] Matthieu Futeral, Armel Zebaze, Pedro Ortiz Suarez, Julien Abadji, Rémi Lacroix, Cordelia Schmid, Rachel Bawden, and Benoît Sagot. mOSCAR: A Large-scale Multilingual and Multimodal Document-level Corpus, June 2024.

[7] Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. Building large monolingual dictionaries at the leipzig corpora collection: From 100 to 200 languages. In *LREC*, volume 29, pages 31–43, 2012.

[8] Xuehong (Stella) He and Aline Godfroid. Choosing Words to Teach: A Novel Method for Vocabulary Selection and Its Practical Application. *TESOL Quarterly*, 53(2):348–371, June 2019.

[9] S. Hunston. Corpus linguistics. In Keith Brown, editor, *Encyclopedia of Language & Linguistics (Second Edition)*, pages 234–248. Elsevier, Oxford, second edition edition, 2006.

[10] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models.* (unpublished), 3rd edition, 2025.

[11] M. G. KENDALL. A NEW MEASURE OF RANK CORRELATION. *Biometrika*, 30(1-2):81–93, June 1938.

[12] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2. Minneapolis, Minnesota, 2019.

[13] Sofie Johansson Kokkinakis and Elena Volodina. Corpus-based approaches for the creation of a frequency based vocabulary list in the EU project KELLY–issues on reliability, validity and coverage. *Proceedings of eLex*, 2011:129–139, 2011.

[14] Tao Lei. *Interpretable Neural Models for Natural Language Processing.* Thesis, Massachusetts Institute of Technology, 2017.

[15] Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding Neural Networks through Representation Erasure, January 2017.

[16] Shaofeng Li, Phil Hiver, and Mostafa Papi. *The Routledge Handbook of Second Language Acquisition and Individual Differences.* Routledge, April 2022.

[17] Pierre Lison and Jörg Tiedemann. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, 2016.

[18] Michael West. *A General Service List of English Words.* Longman, Green and Co., London, 1953.

[19] P. Nation. Vocabulary size, text coverage and word lists. *Vocabulary: Description, acquisition and pedagogy/acquisition and pedagogy*, pages 6–19, 1997.

[20] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang,

Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, C. J. Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. GPT-4 Technical Report, March 2024.

[21] Shahzad Qaiser and Ramsha Ali. Text mining: Use of TF-IDF to examine the relevance of words to documents. *International Journal of Computer Applications*, 181(1):25–29, 2018.

[22] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[23] Anand Rajaraman and Jeffrey D. Ullman. Data mining. In *Mining of Massive Datasets*, pages 1–17. Cambridge University Press, 2011.

[24] Nils Reimers and Iryna Gurevych. Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation, October 2020.

[25] E. Rich. *Artificial Intelligence*. Artificial Intelligence. McGraw-Hill, 1983.

[26] Petr Savický and Jaroslava Hlavácová. Measures of Word Commonness. *Journal of Quantitative Linguistics*, 9(3):215–231, December 2002.

[27] Charles Spearman. Correlation calculated from faulty data. *British journal of psychology*, 3(3):271, 1910.

[28] Jörg Tiedemann and Santhosh Thottingal. OPUS-MT–building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 479–480, 2020.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All You Need. *Advances in neural information processing systems*, 30, 2017.

[30] Giulia Vilone and Luca Longo. Notions of explainability and evaluation approaches for explainable artificial intelligence. *Information Fusion*, 76:89–106, 2021.