Institut für Maschinelle Sprachverarbeitung
Universität Stuttgart
Pfaffenwaldring 5B
D-70569 Stuttgart

Master thesis

How Well Do Language Models Understand Grammar? A Case Study On Japanese

Gerhard Christian Breul

Studiengang:	M.Sc. Infromatik
Prüfer*innen:	Prof. Sebastian Padó
Betreuer:	Dmitry Nikolaev
Beginn der Arbeit:	04.05.2022
Ende der Arbeit:	04.11.2022

Contents

Abstract

Modern attention-based language models such as BERT and GPT have been shown to outperform previous state-of-the-art models on many NLP tasks. This performance implies a level of understanding of grammatical structures. This work attempts to contribute to the growing body of research assessing this understanding, by exploring language models' ability to predict the transitivity of verbs in Japanese, which seems to be somewhat underrepresented in research compared to English. I consider a variety of language models with different architectures, tokenization approaches, training data, and training regimes. In doing so, I find that bidirectional models outperform unidirectional ones, that different types of perplexity calculation can be advantageous in certain situations and should be considered on a case-by-case basis, and that the tested models only gain a somewhat limited understanding of the grammar required for the Transitivity Prediction task.

1 Introduction

Modern language model architectures based on the Transformer architecture proposed by Vaswani et al. (2017) demonstrate an impressive ability to produce and process natural language. Models like BERT (Devlin et al., 2018), GPT (Radford et al., 2018), and XLNet (Yang et al., 2019) have set new standards for a multitude of NLP tasks, such as question answering, translation, summarization, and diverse tasks pertaining to language understanding. To achieve such results, these models have to construct some internal representation of syntactical and semantic information about a given sequence, in order to, for example, extract the correct answer to a given question. For the performance of many NLP tasks, it is likely beneficial to be able to reliably identify parts of speech or dependencies between words, or recognize the topic of a given context. In other words, a machine needs a degree of understanding about the inner workings of a language in order be able to process it effectively.

Indeed, recent research shows that it is possible to extract specific information

about grammatical structures from word embeddings (Hewitt and Manning, 2019; Liu et al., 2019a). However, not all attempts at extracting grammatical knowledge are as successful (Kogkalidis and Wijnholds, 2022). Models seemingly creating representations of some, but not all of the grammar of a language, implies that their understanding of language may be incomplete. What knowledge models learn, and how to extract it, is a topic of ongoing research (Rogers et al., 2020; Lin et al., 2019; Hu et al., 2020). According to this research, not only the type of grammatical structure that is inspected, but also other factors, such as what kind of architecture and training regimen is employed, seem to play a role in the level of grammar understanding a model has (Kim et al., 2020).

English, as the de facto lingua franca of the scientific world, has a big share of research efforts regarding grammatical capability of language models directed towards it. Generally, models appear to be able to grasp grammar of this language relatively well, which may in part be owed to the fact that most models are specifically designed with it as their first priority, and in part to some of the specific attributes of the language itself, such as a strict sentence order and simple conjugation. However, as one might suspect, this does not necessarily translate well into other languages. A different syntax necessitates learning different rules, possibly requiring a more complex architecture. An example of how Transformer-based models fail to understand grammar is given by Kogkalidis and Wijnholds (2022). They show that BERT fails to recognize non-context free patterns of dependencies in Dutch. Many of these analyses inspect a single model on one or more specific tasks. Considering and comparing multiple types of models may yield insights into which model properties affect grammatical understanding in what way.

All this is to say that our knowledge about the limits of what Transformer-based language models can and cannot understand is still quite incomplete. In this work, I assess the grammatical "understanding" of a range of pretrained Transformer-based models.

I take into consideration three generative models that utilize variants of the GPT-2 architecture, and four models based on BERT architecture, three of which are trained on BERTs MLM task, and one which is trained to discriminate "im-

poster" tokens, tokens that have been replaced by a smaller generative model. The selected models represent a range of different training tasks and regimens, different architectures, and tokenization approaches. As the method of determining syntax understanding, I use the prediction of transitivity in Japanese sentences: Models decide, given two Japanese sentences, one with a transitive and one with an intransitive verb, which of the two otherwise identical sequences is more plausible.

This method requires the models to be able to perform three separate basic tasks:

- They need to be able to differentiate between transitive and intransitive verbs.
- They need to determine the number and type of arguments referring to the verb.
- They are required to know the rules according to which each type of verb becomes viable or unviable depending on the number and type of arguments.

I use perplexity as a metric to determine a model's preference for one type of verb over the other. As for some of these models, perplexity of a sequence or part of a sequence can be calculated in multiple ways, I consider a total of 19 combinations of models and methods of perplexity calculations.

I find that model architecture, tokenization approach, and training regimen affect performance on the task selected for evaluation, with unidirectional models performing worse than bidirectional ones. Models' predictions tend to agree most strongly with predictions made by models of the same architecture, with different modes of likelihood calculation within a model only occasionally agreeing less with others of the same model than with architecturally similar but distinct models. An exception to this can be observed in multilingual models. These models, despite using BERT architecture, even if their overall performance is decent (as is the case for XLM-RoBERTa), produce predictions that do not strongly correlate with those of either uni- or bidirectional models, suggesting that the language features they consider are distinct from other models.

This thesis is structured in seven chapters: After this introductory chapter, I will lay the foundation in the Background chapter, followed by a chapter reviewing

related work. After this, the approach used for this research will be detailed in the Methods chapter. This is followed by the presentation and consequent discussion of the results in the following two chapters. Finally, I summarize my work in the Conclusion chapter.

2 Background

In this chapter, I will review two of the research areas this work refers to: Attention-based deep learning language models based on the Transformer architecture, and Japanese grammar, especially concerning transitivity.

2.1 Transformer-based Architectures

Since the publication of the original Transformer Vaswani et al. (2017), language models based on its design have become ubiquitous in NLP. The Transformer architecture consists of an encoder and a decoder, each consisting of originally 6 de- or encoder blocks stacked on top of each other, respectively. One such encoder block is made up of a multi-head self-attention layer followed by a fully connected feed-forward layer. On the decoder side, each block has an additional attention layer with masked attention at the bottom, in effect creating a unidirectional model. In contrast to state-of-the-art models of the time, which were based on convolution or recurrent layers, this model relies completely on attention mechanisms, making it easier to train, while significantly outperforming them on the translation task.

Unsurprisingly, such results inspired the development of models that use the same attention mechanism. Arguably the most well-known of those are the pretrained models BERT (Devlin et al., 2018), which is based on its encoder, and GPT (Radford et al., 2018), which implements its decoder.

(Devlin et al., 2018) adapt the number of encoder blocks, the number of attention heads per attention layer, and the hidden size of the feed forward layer, and propose two architectures: a smaller version with 110M parameters, and a larger one with

340M parameters. These models are trained on two tasks: Masked Language Modeling, also known as the cloze task (Taylor, 1953), and Next Sentence Prediction. For Masked Language Modeling, 12% of tokens are replaced by the [MASK] token, 1.5% of tokens are replaced by a different random token, and 1.5% are not replaced. The model then has to reconstruct the original tokens within the sentence. Replacing some tokens with random ones instead of masks appears to be rather important for performance on downstream tasks, where often the mask token is not used. The Next Sentence Prediction task is the binary decision of, given two sentences A and B, whether sentence B follows sentence A.

BERT received much attention by researchers (Rogers et al., 2020), and many improvements to its training regimen have been proposed. For example, Liu et al. (2019b) find in their own evaluation that the Next Sentence Prediction Task is not conducive to downstream task performance, and therefore drop the objective. Instead, their model, RoBERTa, is trained on more training data and longer sequences. Another avenue of research concerns the acquisition of multiple languages by a single model. Multilingual BERT [1] is a model that uses identical architecture and training objective to BERT-base, but is trained on over 100 languages (102 for the original and 104 for the newer, cased model). While it has weaknesses, especially for low-resource languages (Wu and Dredze, 2020), it performs surprisingly well at cross-lingual model transfer (Pires et al., 2019), given its lack a cross-lingual pretraining objective. Combining the training improvelemts of RoBERTa with a multilingual approach, Conneau et al. (2019) propose XLM-RoBERTa, effectively a multilingual RoBERTa model. This model is trained on one to two orders of magnitude more data per language and seems to generally outperform Multilingual BERT. Clark et al. (2020) propose a different method for improving on BERT's language understanding. Instead of Masked language modeling, their model, ELECTRA, is trained on a Replaced Token Detection task. The generator, a small Masked Language Model, which, similar to BERT but smaller in size, gets sequences with 15% of its tokens masked, which it replaces. The second model, which is architecturally identical to BERT, called the discriminator, now is tasked with determining which

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

of the tokens in the sequence produced by the generator belonged to the original sequence, and which were replaced. As a result of this regimen, ELECTRA requires less training, as all tokens of a sequence are considered, and has less of a disconnect between pretraining and fine-tuning.

While Transformer-encoder based architectures are useful for many applications, given their bidirectional nature, their ability to generate language is limited. For such tasks, a generative language model such as GPT (Radford et al., 2018) or its evolution GPT-2 (Radford et al., 2019) is preferable. These models implement the Transformer's decoder, meaning its defining feature is masking in its attention layers, preventing tokens from attending to left-hand context. GPT-2 builds on GPT mainly by increasing its size (from 117M parameters to 345M parameters for the GPT-2 base model).

The functional difference between the BERT-type and GPT-type models that is expected to be most relevant for understanding grammar is GPT's architectural inability to attend to right-side context as well as BERT's limited ability to consider multiple sentences as a result of its training objective. As such, a slightly altered masked language modeling objective, for example as employed by RoBERTa, may be beneficial to such a model.

2.2 Linguistics

As this work centers around a linguistics problem, I will survey research on the topic of transitivity, which is the main topic of interest, in this section.

2.2.1 Transitivity

Transitivity is an important concept in languages all over the world. The term refers to a property of a clause which describes the transferal of an action from one party, the agent, to another, the patient. Intuitively, the easiest method of determining whether a phrase has high or low transitivity is to observe the number of participants; with a single participant, no action can be transferred, resulting in

low transitivity, while multiple participants indicate some level of action transferal, increasing transitivity. Apart from number of Participants, Hopper and Thompson (1980) identify nine other parameters which affect a clause's transitivity, such as aspect, which considers whether an action is completed, and agency, which considers the ability of the agent to effect the transfer of the action.

Transitivity as determined by agent count often affects the choice of verbs across languages. Generally, some verbs require multiple arguments, while others accept only one. An example in English is the verb *to go*, which only accepts a subject as argument (for example *I go*), making it an intransitive verb, while a transitive verb like *to throw* can take a subject and a direct object, as in *The boy throws a rock*. Often, transitive and intransitive verbs describing the same situation share an origin. To use another simple English example, *I open the door* and *The door opens* share the same verb, even though the first sentence takes two arguments, while the second takes two. This is what is known as a labile alternation pattern. A different pattern is exhibited by the German verb pair *liegen* 'to lie' and *legen* 'to lay sth. down'. Although both words are similar in meaning and share a historic root, the former does not accept a direct object, while the latter requires it. This type of relation, where both verbs share a root and neither is derived from the other, is known as a equipollent pattern.

Haspelmath (1993) looks at verb pairs from 21 languages, and finds stark differences between languages in the way transitive and intransitive verbs are derived: Some languages, such as Greek, German, and English, have a strong preference for labile patterns, using the same verb in clauses with different transitivities. Others, such as Russian and Romanian, tend to use an anticausative pattern, meaning the intransitive verb is derived from the transitive one. In the case of Japanese, the majority of verb pairs display an equipollent pattern. However, causative (where the transitive verb is derived from the intransitive) and anticausative derivation patterns do also occur. Importantly, labile constructions are effectively absent form Japanese, meaning that every verb is either transitive or intransitive. A pair of verbs with differing transitivity, where both alternants share an origin, is called a transitivity pair.

2.2.2 Japanese

The Japanese language has aspects that set it apart form others and which need to be addressed. Likely the most obvious of those is its writing system. Japanese uses three distinct scripts: hiragana, katakana, and kanji. Hiragana and katakana, together known as kana, are syllabaries which essentially denote the same syllables. The difference between the two lies in what they are used for: Hiragana are used for inflections of verbs and for words with a grammatical function like particles, as well as some native Japanese words. Katakana are mostly used to transcribe non-Chinese foreign loanwords into Japanese. Each of these scripts encode around 100 unique syllables, with which any Japanese word can be spelled out. Finally, kanji are logographic characters originating from Chinese. Rather than syllables, each of these characters represents a meaning and usually has multiple ways to be read. As an example of Japanese script, we consider the sentence '(I) open the door':
ドア を 開ける。
doa wo a-keru
The first two symbols (ドア) are katakana and spell the English loan word 'door'. The next symbol (を) marks the direct object and as such, is a hiragana character. Lastly, 開ける is the verb of the sentence. It consists of the kanji 開 as its stem, which describes the action of opening, and the inflection written in hiragana. Hiragana used for inflection of verbs and adjectives in this manner are called okurigana.

I will avoid the use of Japanese script wherever possible for the sake of simplicity. Instead, to make Japanese readable, it will be written in romaji, a transcription of the Japanese script using Latin letters, used for example for typing. Since romaji directly translates each kana into a sequence of Latin letters, there are some differences to the notation usually used in literature, which transcribes the reading of words, instead of their spelling. For example, Japan's capital Tokyo is written as *Toukyou* in romaji, as a written *ou* is usually read as a long *o*. Another such case is the accusative marker, which as romaji is written as *wo*, while usually (but not always) read as *o* in modern Japanese. Given that exact pronunciation is not the focus of this work, using romaji instead of a transcript based on phonology, as is commonly used in

linguistics literature, should be sufficient.

As stated in the section about transitivity, Japanese strictly distinguishes between transitive and intransitive verbs, and usually derives both from a common root. For Japanese text, this root in most cases is denoted by the kanji stem of a verb, with the okurigana differentiating transitive from intransitive. The alternants of transitivity pairs therefore tend to look quite similar to each other, and there is no general rule to decide a verb's transitivity purely based on its okurigana, without additional lexical information. In terms of the structure of simple sentences, the language has verb-final order. Apart from this, sentence structure is relatively free, although subject-object-verb is considered the standard. Instead of by position, argument types are conveyed by particles, markers which are appended to the noun phrase. The *ga* particle marks the subject of a verb. In terms of transitivity, this denotes the agent from whom the action originates. The *wo* particle marks the direct object of a transitive verb. In certain situations, it is acceptable to leave out such particles (Minashima, 2001). However, this phenomenon is mostly observed in the spoken language, and a short preliminary experiment suggested that BERT does not have a good understanding of it, which is unsurprising, given its training dataset was Wikipedia. The *ha* particle is a so-called topic marker. An argument with this marker can, depending on the context, take the role of either subject or object. Like arguments, adverbial phrases can also be placed at any point before the verb. Apart from a few exceptions, relative clause construction in Japanese uses a gap strategy (Comrie and Polinsky, 1993), where the relative clause is a basic clause structure with a missing argument, prefixed to the head noun, which fills the role of the gap.

Another feature of Japanese is that it is a pro-drop language, meaning that pronouns are regularly omitted. This creates not only an obvious difficulty in determining presence or absence of arguments, but also for training, as zero anaphora constructions may trow language models off. Umakoshi et al. (2021) propose a method to alleviate this problem by using parallel text of a language that is not pro-drop, as in such languages, the dropped argument will usually be explicitly named. The pro-drop tendency combined with the lack of verb conjugations based on grammatical person and number has the added effect of making Japanese uniquely sensitive

to context, as for example an omitted subject can often not be determined without it.

Languages such as Japanese and Chinese also set themselves apart by their lack of spaces to mark the beginning and end of words. For NLP, this mainly complicates tokenization, which requires more sophisticated means of recognizing words. This aspect and its effects will be examined more closely in coming chapters.

An example of a Japanese grammar is given by Martin (2003).

3 Related Work

In this chapter, I will go into some of the existing research concerning neural language models, especially with regard to their capacity for syntactical understanding.

On the topic of syntactical understanding, a growing body of research can be found, which, as already eluded to in the introduction, mainly focuses on English models and syntax. The approaches used in this research can roughly be categorized into one of two types: Behavioral approaches, which evaluate a model's outputs given specific inputs, effectively considering it as a black box, and probing approaches, where outputs of a model's layers are evaluated using a so-called probe, which takes a specified layer's output as input features to make a prediction. By training such a probe in a supervised fashion, one can extract specific information encoded in those layers.

3.1 Grammatical Knowledge

A comprehensive overview of the research done on BERT is given by Rogers et al. (2020). This overview collects proposed improvements and insights gained from more than 150 studies, finding generally applicable statements about the extend and limitations of BERTs syntactic and semantic knowledge. It also introduces many of the training regime improvements that have been suggested, such as RoBERTa (Liu et al., 2019b), which drops the next sentence prediction task and increases

the amount of training data by an order of magnitude, and XLNet (Yang et al., 2019), which instead of masking, uses word order permutation. While this analysis mainly concerned with research on monolingual English BERT, and its findings are therefore not guaranteed to be applicable to other languages or multilingual models, it serves as an important overview of what one should expect BERT to be capable of.

Investigating computational limitations, Bhattamishra et al. (2020) explore the ability of transformer-based architectures to recognize formal languages. They find these models to be limited in their ability to recognize certain types of regular languages, as compared to LSTMs. Bai et al. (2021) show that it is possible to improve BERT's and RoBERTa's performance on downstream tasks by training attentions to reflect syntax trees. This implies that where data annotated with such information is available, it is beneficial to utilize this kind of partially supervised training approach.

3.1.1 Probing Approaches

A probe, as mentioned above, often refers to a single (trained) linear transformation or a small neural network consisting of a one or a few feed-forward layers. It takes the output of a certain layer of the model as its input and is then trained to make task-specific predictions based on this. Probes are commonly used to extract encoded information from different layers of models, as a method of finding linguistic knowledge in these encodings.

Peters et al. (2018) probe different types of bidirectional models - an LSTM, Transformer, and CNN - for grammatical and contextual knowledge using an array of diverse tasks, such as POS Tagging and constituency parsing. Their results show that for many of the posed tasks, good predictors can be found within a model's layer outputs, with differing tasks' most optimal representations being found within different layers of the model. It should be noted that the implementation of the transformer model Peters et al. (2018) used, while bidirectional, does not closely resemble BERT, but rather a very small forward- and backward GPT, as it utilizes

a Transformer decoder rather than an encoder.

In similar experiments, Jawahar et al. (2019) probe BERT for syntactic and semantic information and find that syntactic information is generally encoded in the middle layers, with semantic information being encoded further up. Tasks requiring the processing of long-range dependency information such as verb-subject-agreement are found to require higher layers.

Such results are corroborated by Lin et al. (2019), who find that BERT encodes positional information in early layers, with deeper layers encoded information becoming increasingly complex. A similar approach is used by Liu et al. (2019a). They compare performance of transformer-based models' (namely BERT and GPT) layer-wise best-performing probes to state-of-the-art task-specific models and find that performance is competitive for many, but not all tasks. The authors determine this to be due to the models requiring more precise data for these tasks.

Hewitt and Manning (2019) extract distances between layer token representations of BERT and ELMo. Using these distances, they attempt to recreate the sentence's parse tree. They observe that these trees can indeed be reconstructed relatively reliably from the distances calculated from embeddings extracted at middle layers of the models (layer 7 and 8 for 12-layer BERT), and increasingly less reliably from distances extracted from lower and higher layers. The implication is that the models in question do construct representations of syntax at certain points within them.

Mueller et al. (2022) compare multilingual BERT and XGLM, a multilingual auto-regressive language model, as well as multiple monolingual models with regard to subject-verb-agreement encoded in their neurons, and find that multilingual models share syntax-sensitive neurons across languages, with XGLM sharing more than multilingual BERT, and that auto-regressive models encode knowledge in a distinct fashion from masked language models.

What knowledge requires specialized training to get encoded also varies from language to language: Koto et al. (2021) explore the document-level information encoded in Spanish, Chinese, German, and English models. They find that while

the point at which the models encode this information tends to be a similar layer across languages, the quality can vary significantly, implying that tasks that are difficult in one language may be simpler in another.

An example of a difficult task is given by Kogkalidis and Wijnholds (2022), who probe Dutch BERT's output layer for its knowledge of certain non-context free constructions. They find BERT largely unable to model such structures. Their results also point towards a possible reason for BERT's performance discrepancies between English and other languages, as the investigated structures would be ungrammatical in English.

In a similar fashion, Ueda et al. (2020) perform cohesion analysis on Japanese text by training a probe on top of BERT's output. While their approach outperforms the previous state of the art on multiple tasks, results are likely still far from human scores.

3.1.2 Behavioral Approaches

Another method of evaluating linguistic knowledge in models is to simply observe their behavior. In the case of masked models, one intuitive method to do so would be to mask a certain word and compare probabilities of grammatical versus ungrammatical tokens. An example of such an approach is demonstrated by Goldberg (2019), who assesses BERTs grammatical knowledge by masking a token and comparing its likelihood with that of a tokenwhich is similar in most aspects, but which violates a certain syntactical rule, such as subject-verb-agreement. For example, given the sequence "The medicine [MASK] an effect", one could compare the likelihood of "has" to that of "have" in the position of the masked token. This effectively forces the model to make a decision between a grammatical and an ungrammatical sequence. According to the author, BERT performs surprisingly well in this setting.

This approach, however, is restricted to specific models and situations. In order to compare a wider range of models, a universally applicable approach is required. Warstadt et al. (2020) construct BLiMP, a dataset of minimal pairs to evaluate models' ability to answer a multitude of syntactical questions. Specifically, this is

achieved by comparing probabilities of two minimally different sentences, where one sentence is ungrammatical, similar to Goldberg (2019). The sequence with the higher probability is considered to be the model's prediction. Transformer-based models outperform the LSTM and 5-gram comparisons on most tasks, with GPT-2 even reaching performance comparable to humans on some of them. Using probability of a sequence, while intuitive and effectively applicable to any language model, comes with its own problem: sequence probability, as the product of each token's probability, is highly dependent on sequence length, meaning that if one sequence is longer that its counterpart, it is highly likely to have a lower probability, regardless of grammatical acceptability.

Commonly used methods to evaluate downstream performance of models are benchmarks such as SQuAD and GLUE: SQuAD (Rajpurkar et al., 2016) and SQuAD 2 (Rajpurkar et al., 2018) are datasets to evaluate Question answering performance, by supplying the model with a paragraph of text about which it is then asked to answer a question. SQuAD 2 introduces the possibility of unanswerable questions making the task more difficult. GLUE (General Language Understanding Evaluation) (Wang et al., 2018) is a test suite collecting multiple tests to quantify language understanding, such as a test for linguistic acceptability, where models are asked to decide whether an English sentence is grammatical (Warstadt et al., 2019), sentiment analysis (Socher et al., 2013), and question answering.

Xiang et al. (2021) create CLiMP, a Chinese language understanding benchmark based on minimal pairs. They compare BERT, multiple LSTM models, and 5-gram-models. while BERT outperforms its competitors with 81.8% accuracy by a wide margin, it is still far away from the 95.8% human agreement. Predictions are determined analogously to BLiMP. More recently, Song et al. (2022) generally confirm the findings of Xiang et al. (2021) by creating another minimal-pair-based Chinese Evaluation dataset named SLiNG, and improving on some of the aspects of CLiMP that were seen as problematic. They expand the scope of models under consideration to include a wide array of mono- and multilingual transformer models. Furthermore, instead of likelihood, they use perplexity (or pseudo-perplexity, if applicable) as the deciding metric. This was one of the criticisms leveled at the prior approach, as

likelihood is by design extremely sensitive to sequence length, which varied in some sentence pairs in CLiMP. Chinese monolingual BERT, as the best-performing of the tested models, reaches an accuracy of 84.8% on this set, which is still significantly below the 97.1% average accuracy of the human control. Interestingly, multilingual models seem to perform especially poorly on both of these Chinese benchmarks.

3.2 Perplexity

As mentioned above, likelihood can be a problematic as a metric because of its dependence on sequence length. Perplexity, as the negative inverse log likelihood normalized by sequence length, is, at least in theory, not correlated to length. Given this property, the metric is commonly used to judge a model's preference of one sequence over another.

Salazar et al. (2019) argue that, for bidirectional models like BERT, calculation of likelihood and perplexity need to be slightly adjusted, as the original formulas only consider one-sided context. To calculate a metric mirroring perplexity, they propose to sequentially mask each token in order to calculate the sequence likelihood which is then used to compute what they call "Pseudo-Perplexity". Given that this process requires as many runs of the model as there are tokens in the sequence, it is somewhat computationally expensive compared to perplexity computations on other models. Therefore, the authors propose perplexity calculation without masking, based on the likelihoods BERT assigns to each token, which only requires one iteration, as a trade-off between task performance and computing requirements.

In this work, when the term perplexity is used in the context of masked language models, it refers to what Salazar et al. (2019) call pseudo-perplexity, unless otherwise specified.

An example for the use of perplexity for grammar understanding assessment is given by Marvin and Linzen (2018): Comparing perplexity of RNNs on minimal pair sentences that differ in syntactic correctness they find that at least according to this metric, such models do not display much grammatical knowledge. Miaschi et al. (2021) investigate the variables that affect perplexity scores for GPT-2 and

BERT. They conclude that the two models react to similar, but distinct aspects of sequences. This implies that these models may learn syntax differently due to their architectural differences.

Lee et al. (2021) propose to use perplexity in the context of fact-checking. To do so, they give one or more sentences of evidence for some fact and a claim, which either agrees or disagrees with the evidence, to a pretrained BERT and GPT-2 model. They then measure the perplexity of the evidence. If it is above a certain learned threshold, the claim is considered false, else it is considered true. This approach to fact-checking performs well in a few-shot-setting compared to models fine-tuned on this task. GPT-2 outperforms BERT on this task, which the authors assume to be due to the difference between perplexity and pseudo-perplexity , but may also be due to pretrained BERT not having any pretraining objective that requires consideration of more than two sentences.

Wang et al. (2022) discuss flaws of perplexity as a metric: They find that while it is supposed to be a measure of linguistic acceptability, it is influenced by factors such as sequence length, repetition, and punctuation marks. This should be kept in mind when planning to use the perplexity metric as decider. Most approaches get around this by using perplexity to force a decision between two minimally different sequences. Such sequences should only differ in one feature in order to eliminate as many sources of difference in perplexity as possible. One should avoid differing punctuation marks or repetitions, and differences in length should be kept to a minimum. The described effect of perplexity decreasing with increasing sequence length can also be observed on the dataset used in this work, for at least three different language models.

4 Methods

In this chapter, I will elaborate on the approach used determine grammatical understanding of current language models.

4.1 Defining "Understanding"

The Goal of this work is to determine the capacity of modern language models to "understand" grammar. To do so, we first have to find an agreeable on a definition of understanding. There may be a discussion to be had about whether an artificial model can truly understand anything without being conscious. For the purpose of this work, however, this view is somewhat impractical, as consciousness is a notoriously difficult concept to define. Understanding, in the context of this work, thus should not presuppose consciousness. On the other hand, most would likely agree that guessing the right answer to a question based only on statistical analysis, for example as exemplified by n-gram models, does not constitute understanding, in the same way that a student learning all answers to a math quiz by heart, instead of calculating each according to a certain set of rules, can not be said to have understood these rules. Distinguishing between learned rules and statistics requires some ingenuity (Anil et al., 2022). Considering this, understanding, for a language model, should ideally involve learning abstract rules from the training data and being able to apply them to unseen data. For this work, I will define understanding as such:

A language model can be said to *understand* a grammatical rule, if it can reliably decide whether a given sequence fulfills it.

4.2 Selection of a Grammatical Rule

To evaluate grammatical understanding, a specific rule or set of rules needs to be selected. I decided to use the rules governing the viability of transitive vs intransitive verbs in simple Japanese sentences for this purpose. I will call the associated problem of deciding whether given a certain context in Japanese, a transitive verb or its intransitive counterpart is more appropriate, **Transitivity Prediction**. The decision to use Transitivity Prediction as a means to assess grammatical understanding is motivated by some of the aspects of the language and the problem itself:

- The verb-final sentence structure in Japanese allows unidirectional models to evaluate the verb in a sentence with a similar amount of information about

the context as bidirectional models. The only part of the context the verb in a unidirectional model is unable to attend to is the sentence-ending token.

- Having transitivity pairs that share a stem and meaning helps to reduce the difference between transitive and intransitive sentences, potentially reducing the effect of co-occurrence biases and other unwanted sources of noise, as such effects will usually be present in both sentences. In some cases, transitive and intransitive sentences differ by as little as a single token.

- Since Japanese has a relatively free sentence order, apart from the verb generally being bound to the end of a sentence, big datasets can be constructed using relatively few components by changing the order of constituents. Furthermore, this allows us to observe how architectures deal with sentence structures that are in this respect more complicated than those found in languages with stricter restrictions, such as English.

- Determining the actual viability with regard to Transitivity Prediction is as simple as counting the number and type of arguments: If a sentence has two arguments or one argument that is a direct object, it is obligatorily transitive; else, both verbs are grammatically acceptable.

Of course, there are also some aspects of the language that pose challenges: In Japanese, as opposed to many European languages such as English or German, ellipsis of arguments is very common. In such cases, the missing argument is typically implied in the context. As a consequence, even in sentences that have, for example, no specified direct object, such an argument might still be implied. Importantly for us, this means that sentences with a single argument or even no specified arguments are not automatically intransitive. Thus, while one can construct a sentence that requires a transitive verb by supplying a direct object to the sentence, there is no method to construct a definitively intransitive sentence, since there is always the possibility of an implied direct object. Another challenge of working with Japanese is the lack of spaces as word delimiters. Tokenizers have to use more sophisticated methods to separate text than simply considering each substring between two spaces

as a word. For Japanese, successful models employ a grammatical parser that uses a dictionary to identify words, the output of which is then further separated according to the WordPiece algorithm. However, if no language-specific specific information such as a dictionary, is used, as may be the case with for example some multilingual models, more general tokenization strategies can lead to problems. An example of such a problem is described in the the subsection regarding XML-R, within the next section.

4.3 Selection of Language Models

Eight pretrained language models based on state-of-the-art architectures available on huggingface.co were selected for analysis, with the goal of covering a wide range of parameter combinations, such as architecture, training data, training approaches, and tokenization approaches. Five of these are based on BERT architecture. The other three are variants of GPT-2 in differing sizes. In this section, I will go over the specifics of each model. Later on, I will refer to each model by the name in brackets instead of its full name for convenience.

4.3.1 BERT Architecture

- Japanese whole word masking BERT (BERT): [2] This model uses the same architecture as BERT-base (Devlin et al., 2018), meaning it has 12 sets of an attention layer followed by a fully connected feed-forward layer. Each attention layer consists of 12 attention heads, while each feed-forward layer has a hidden size of 768. This results in a model with 110 million parameters. For tokenization, it uses the MeCab morphological parser (Kudo, 2005), an open source text segmentation library for Japanese, with the IPA dictionary, to segment text into words, and then uses the WordPiece algorithm to segments them further, resulting in a vocabulary size of 32000. The training dataset for

[2] can be found at huggingface.co/cl-tohoku/bert-base-japanese-whole-word-masking, commit ab68bf4, downloaded 26.05.2022

this model is Japanese Wikipedia, using the dump from September 1, 2019. While it is not specifically stated by the creators, WordPiece is likely trained on the same dataset.

- Japanese character based BERT (cBERT): [3] Architecturally, this model is identical to BERT. Training was done on the same dataset as BERT as well. Its unique aspect (among the selected models) is its tokenization: According to its description, this model also uses MeCab with IPA to segment text into words, but then further separates these words into single characters, resulting in a character-based tokenizer. Any word that is not recognized by the MeCab parser is considered an unknown token, and will not be considered for further tokenization. This result is that rare kanji not occurring in words from the dictionary do not find their way into the vocabulary. Another, slightly different character based BERT model pretrained by cl-tohoku is also available on huggingface: this model uses Unidic 2.1.2 instead of the IPA dictionary, resulting in a bigger vocabulary of 6144, compared to the original size of 4000. A small preliminary experiment showed both models performing similarly well on a transitivity prediction task. With the first model outperforming the second by a small margin, I decided to only consider the first for further analysis, in order to keep the number of models being analyzed manageable.

- Multilingual BERT (mBERT): [4] This model once again uses the same architecture as those above, but was trained on the Wikipedia datasets of 102 different languages according to its description on huggingface (this number differs from the 104 languages cited by Pires et al. (2019)). To process Japanese text, this model represents each kanji as its own token, as a character-based model would, but uses WordPiece for non-kanji symbols. This results in somewhat of a hybrid character- and WordPiece-based tokenization approach.

- XLM-RoBERTa-base (XLM-R): [5] (Conneau et al., 2019) This multilingual

[3] huggingface.co/cl-tohoku/bert-base-japanese-char, commit 6aa4c7b, downloaded 26.05.2022
[4] huggingface.co/bert-base-multilingual-uncased, commit 800c34f downloaded 07.09.2022
[5] huggingface.co/xlm-roberta-base, commit f6d161e downloaded 17.09.2022

model uses the same architecture as the models above. It is, however, trained on a much bigger dataset than the previous models: in order to learn 100 languages, apart from Wikipedia, it also uses CommonCrawl data which has been cleaned following Wenzek et al. (2019), increasing the amount of training data per language by two orders of magnitude on average. Just like its namesake RoBERTa, this model forgoes the next sentence prediction task used in the training of BERT. Instead, it uses consecutive sentences cropped to the maximum sequence length in training. It uses the SentencePiece algorithm for tokenization, which is useful for a multilingual model, as it does not require language dependent logic, but, in the case of Japanese, creates a problem for language understanding: Due to the lack of spaces as word delimiters, SentencePiece sometimes produces erroneous tokenizations. It occasionally separates sequences in such a way that tokens end up containing parts of multiple words. A crucial example of this misbehavior for our application is its tendency to fuse particles with the stems of verbs, especially if those verbs co-occur with the particles and are otherwise not very common. This becomes particularly noticeable with the "wo" particle, since the symbol for this particle does not appear in any other context. Such behavior is problematic because a token containing a particle and part of a verb may obscure the underlying words to the model, making it difficult to determine their properties. Such a token can not easily be categorized as either the stem of a verb of a certain type, nor as a particle, but will likely constitute its own category of word to the model. Ultimately, such inconsistencies could impair the model's ability to create an internal representation of a sequence's grammatical structure during training as well as evaluation, resulting in reduced performance.

- ELECTRA base Japanese discriminator (ELECTRA): [6] This model is the discriminator part of an ELECTRA base model as described by Clark et al. (2020), trained on the Japanese Wikipedia dump from June 1, 2021. Instead of masked language modeling, the ELECTRA discriminator is trained to rec-

[6] huggingface.co/izumi-lab/electra-base-japanese-discriminator, commit 9a50f72, downloaded 12.08.2022

ognize whether a token has been replaced by a smaller-scale masked-language-model, in this context called the generator. In training, 15% of tokens are masked and then predicted by this generator. If the generator prediction differs from the original token, this replaced token is considered an imposter, if the prediction is correct, it is considered not an imposter. Owing to this pretraining regimen, the probabilities for whether or not any given token is an imposter is expected to be below 15%. One potentially beneficial effect of this approach is that the discriminator does not require a mask token in training, making it easier to fine-tune the model for downstream tasks which do not require it, as there is no discrepancy between (masked) pretraining and (maskless) fine-tuning. Another effect is a drastic reduction in training time, since the model can effectively use all tokens of a sequence for training, instead of being limited to masked tokens. The discriminator uses the same architecture as BERT, while the generator has one third of the size. The ELECTRA model I am evaluating uses an approach to tokenization almost identical to the Japanese BERT model: MeCab using the IPA dictionary for word tokenization, WordPiece for subword segmentation. The difference between these tokenizers is ELECTRAs slightly bigger vocabulary of size 32768.

4.3.2 GPT-2 Architecture

- Japanese GPT-2 (GPT-2): [7] Representing a typical mid-sized GPT-2 architecture, this model, which is the most popular monolingual Japanese language model of this specific type on huggingface, was selected. GPT-2 models like this one implement the decoder of the Transformer architecture proposed by Vaswani et al. (2017) in contrast to the Transformer-Encoder-based BERT. Specifically, this model consists of 24 Transformer decoder layers, each of which are themselves made up of a masked attention layer with 16 attention heads and a feed-forward-layer with a hidden size of 1024, resulting in a significantly bigger model than BERT with 345M parameters. It was trained on Japanese

[7] huggingface.co/rinna/japanese-gpt2-medium, commit f464b76, downloaded 12.04.2022

CC-100 (Wenzek et al., 2020), a 15GB CommonCrawl dataset, and Wikipedia, with a causal language modelling objective. The exact Wikipedia version is not named, but Git changelogs point towards the August 1, 2021 dump [8]. Tokenization is done using a SentencePiece tokenizer that was trained on the same Wikipedia dataset. As explained in the section on XLM-R above, this tokenizer shows some unintended behavior which could impact performance negatively.

- Japanese GPTneo (GPTneo): [9] This model uses the same tokenization as GPT-2, with the only difference in architecture being an increased hidden size of 2048, making it the biggest model out of all those selected by a small margin, with around 350M parameters (even though the models full name, gpt-neo-japanese-1.3B, would suggest a much bigger number). The training objective is identical to GPT-2, and GPTneo was trained on the same datasets in addition to Japanese OSCAR (Suárez et al., 2019; Ortiz Suarez et al., 2020), a large dataset obtained by language classification and filtering of the Common Crawl corpus.

- Japanese GPT-2 small (GPT-2 small): [10] This model implements the small version of GPT-2, with 12 instead of 24 layers, 12 instead of 16 attention heads per attention layer, and a hidden size of 768 instead of 1024, resulting in 117M parameters, effectively bringing its size in line with BERT. Unlike the two other GPT-2 based models, this model uses simple byte pair encoding without a dictionary for tokenization. In practice, this bypasses SentencePiece's problematic behavior, at least for the situation described, where SentencePiece considers a verb stem and a preceding argument marker to be one word, on the constructed datasets being used. Training was done on Wikipedia and a subset of the Japanese CC-100 dataset, although details on this subset are not specified.

[8] https://huggingface.co/rinna/japanese-gpt2-medium/commit/ae4875affd0259f0cd8debaea23174fc524c05df
[9] huggingface.co/yellowback/gpt-neo-japanese-1.3B, commit 69add76, downloaded 01.09.2022
[10] huggingface.co/ClassCat/gpt2-base-japanese-v2, commit 52e7199, downloaded 04.09.2022

4.4 Dataset Generation

To generate sequences to evaluate these four models on, I construct simple sentences consisting of a verb at the end, and up to three further constituents, which can be either arguments or an adverbial.

4.4.1 Verb Selection

To select a suitable set of verbs, I start out with a list of 306 verb pairs compiled by Kageyama and Jacobsen (2016). The pair *hairu* "to enter" - *ireru* "to put in" is added manually by me. This specific pair is missing in the original list because the verbs are derived from different stems, as indicated by the stems' differing readings. However, in writing, these verbs use the same root kanji. Since our models can only consider the written form, this pair appears like many other pairs on the list to them. I then filter pairs from this list according to certain criteria:

- if according to the JMdict dictionary (Breen, 2004), one or both alternants are usually written using kana alone. For such pairs, models would be much less likely to recognize the verb in its kanji form.
- if both alternants share the same transitivity, meaning the list and JMDict conflict.
- Differing root kanji, as part of the idea behind Transitivity Prediction is to make the difference between sentence pairs as minimal as possible, which includes differing verb stem writings.
- if an alternant is not present in the dictionary at all.

For the remaining pairs, I record their number of occurrences in the Wikipedia corpus. This corpus is relevant because as shown above, it is a common training set for all models under evaluation. Verb pairs where the number of occurrences of one alternant is zero are disregarded as well, as in these cases, a model trained on Wikipedia did not have had an opportunity to learn the transitivity of a verb. After

this filtering, 225 out of the original 306+1 verb pairs remain candidates for the dataset. Of these 225 verb pairs, the 50 most common ones are selected for the main dataset. I determine how common a verb pair is by the number of occurrences of its less common alternant in the Wikipedia dataset.

Using these verb pairs for an evaluation dataset ensures that every model has had sufficient opportunity to learn the relevant grammatical properties of these verbs. One caveat here is that the Wikipedia datasets that were used for training are not identical, as they were created from different dumps at different points in time.

Presumably, more common verbs would allow the language model to learn whether or not a verb is transitive with more certainty, thus improving performance on a task where the objective is to determine if a sentence should have a transitive or intransitive verb. For Transitivity Prediction, recognizing transitivity of a verb is a necessary, but not a sufficient condition. The model also needs to recognize the number and type of dependent arguments, and understand that depending on these arguments, an intransitive verb may be ungrammatical. Since abilities concerning arguments are independent of the verb, more knowledge about the verb itself is not expected to improve them. Therefore, while more common verbs are expected to induce better performance on transitivity prediction on average, the value of a model's verb knowledge would plateau at a certain point, after which the ability to recognize arguments likely becomes the deciding factor for improvement.

4.4.2 Selection of Arguments and Adverbs

Apart from verbs, arguments and adverbials are required for the evaluation dataset. The arguments were selected to cover a wide rage of attributes, from personal pronoun to intangible concept: *watashi* 'I', *kare* 'he', *hito* 'person', *kangofu* 'nurse', *neko* 'cat', *tori* 'bird', *ongaku* 'music', *jiyuu* 'freedom', *isu* 'chair', *zairyou* 'ingridients', *doa* 'door'. The first two arguments are definite personal pronouns. The second pair are nouns representing persons, making them are no longer necessarily definite. *Neko* 'cat' and *tori* 'bird' are animate but no longer describe a person. The last five nouns represent inanimate things or concepts, with *isu* 'chair', *zairyou* 'ingridients', and

doa 'door' being inanimate and tangible, while *ongaku* 'music' and *jiyuu* 'freedom' are intangible. These nouns and pronouns, combined with one of three particles (*wo*, *ha*, *ga*) make up an argument. Lastly, there are three adverbial phrases that are used to create the dataset: *Katteni* 'voluntarily, on its own accord', *yoku* 'well, often', and *tabun* 'probably'. Sentences are formed by prefixing any combination of zero or one adverbial and 0 to 2 differently marked arguments in any order to a verb.

4.4.3 Sentence Construction

Now that all components have been selected, sentences can be constructed from them. By creating all possible combinations of verbs, arguments and adverbials that can be built in this manner, we arrive at a dataset with 374750 transitive and 374750 intransitive sentences. Sentence construction can be described using a context-free grammar:

$$
\begin{array}{lll}
S \to C + V + . & C \to \{\} & (AW) \to AW \\
C \to (AGH) & (AGH) \to (AG)H & (AW) \to WA \\
C \to (AGW) & (AGH) \to (AH)G & (GH) \to GH \\
C \to (AHW) & (AGH) \to (GH)A & (GH) \to HG \\
C \to (AH) & (AGW) \to (AG)W & (GW) \to GW \\
C \to (AW) & (AGW) \to (AW)G & (GW) \to WG \\
C \to (AG) & (AGW) \to (GW)A & (HW) \to HW \\
C \to (GH) & (AHW) \to (AH)W & (HW) \to WH \\
C \to (GW) & (AHW) \to (AW)H & V \to verb \\
C \to (HW) & (AHW) \to (HW)A & A \to adverbial \\
C \to A & (AG) \to AG & G \to noun + ga \\
C \to H & (AG) \to GA & H \to noun + ha \\
C \to W & (AH) \to AH & W \to noun + wo \\
C \to G & (AH) \to HA &
\end{array}
$$

Capitals denote non-terminals, and lower case words are stand-ins that denote one of the associated set of terminals.

This dataset will be called the *base* dataset, in order to distinguish it from alternative datasets described next.

4.4.4 Alternative Datasets

Apart from the main dataset, three smaller datasets were created to further explore certain aspects that may affect Transitivity Prediction, the first of which was already briefly mentioned:

- **Rare verbs** : For this dataset, I utilize the 50 least common verb pairs of the filtered verb pair list, instead of the 50 most common, creating a dataset of equal size. This dataset is intended to indicate the effect of having less training data on the task. Generally, I expect to see decreased performance for the *rare verbs* dataset, especially for those models that were trained exclusively on the Wikipedia corpus.

- **Relative clauses**: This dataset introduces relative clauses to arguments. Specifically, I randomly sample 10000 sentence pairs form the base dataset and prefix one of three generic relative clauses to one of its arguments. I exhaustively generate all sentences that can be created in this manner, thus creating six new sentence pairs for sentences with two arguments, and and three new sentence pairs for sentences with one argument. This results in a dataset with 59067 sentence pairs. The three generic relative clauses are:
  kinpatsu no onna ga shiranakatta Arg
  'the Arg which the blonde woman didn't know'
  kinpatsu no onna wo tasuketa Arg
  'the Arg which helped the blonde woman'
  kinpatsu no onna ga ki wo tsuketa Arg
  'the Arg the blonde woman was wary of'

  The intended effect of modifying the sentences like this is increased structural complexity and the introduction of particles not related to the main verb of the sentence. The first sentence introduces a subject-marking ga-particle into

the sentence, the second introduces an object-marking wo-particle, and the last one introduces both. Given the additional markers, a model that relies on co-occurrence of particles with certain types of verbs for predictions is expected to perform worse on this dataset. For good performance, the model needs to be able to differentiate between particles that belong to a relative clause and those that do not. These specific clauses were selected for their relative neutrality as well as to introduce the different particles, while working with any kind of argument, be it animate or inanimate, tangible or intangible, without becoming nonsensical.

- **Longer sentences**: Any difference in predictive ability on the *relative clauses* dataset compared to the base dataset might not be the effect of the relative clauses themselves, but of the elongated input sequence. To isolate the effect of a longer sequence without a relative clause, I add a neutral phase to the beginning of each sentence in the 10000-sentence-pair sample used for the construction of the relative-clause-dataset. The phrase used is:
  utsukushii tenki na hi, tokidoki...
  'on a day with beautiful weather, sometimes...'

4.5 Prediction of Transitivity

To decide whether a model prefers the transitive or intransitive alternant of a verb in a given sequence, I compare the perplexity of those sequences to each other. This metric is useful for our application because it takes into account token likelihoods while theoretically being independent of the number of tokens. Technically speaking, perplexity is the inverse likelihood of a sequence, normalized by its length. As such, it can be thought of as a metric describing how acceptable the language model deems a sequence to be, with low perplexity values corresponding to high acceptability. Perplexity by itself, however, has been shown to not necessarily be a good predictor of grammatical acceptability, as factors such as sequence length can have an effect on it (Wang et al., 2022). In fact, my own results confirm that sequence length has a

negative correlation with perplexity for the base dataset as well. While this finding can not be generalized as the dataset only reflects very specific, simple Japanese constructions, it is something to be kept in mind. A sequence might have a higher perplexity value than another merely due to the fact that one argument rarely occurs in the proximity of the other, but since arguments are chosen independently of the verb, this change in perplexity has nothing to do with the actual transitivity of the sentence. Thus, I compare the perplexity value of one sentence to that of a minimally different sentence, with the only difference between them being the verb. This ensures that any bias affecting the perplexity of one sentence is likely also affecting the perplexity of the other.

Given the sequence X of length n consisting of tokens $x_1, x_2, ..., x_n$, perplexity can be described by the following formula:

$$PP(X) = \sqrt[n]{\prod_{i=1}^{n} \frac{1}{L_{model}(x_i, X)}}$$

where $L_{model}(x_i, X)$ is the likelihood assigned by the model to token x_i given the sequence X. Thus, to compute perplexity, it is necessary to extract these likelihoods form the models we want to investigate. Due to differences in architecture and training objective, these values will have differing interpretations from model to model.

For the BERT-based masked language models, meaning BERT, cBERT, ml-BERT, and XLM-R, the likelihood of a token in a sequence represents the probability the model assigns to the token if it is masked:

$$L_{BERT}(x_i, X) = P_{BERT}(x_i | X_{masked})$$

With X_{masked} representing the sequence X with token x_i replaced by a mask token:

$$X_{masked} = (x_1, ...x_{i-1}, mask, x_{i+1}, ...x_n)$$

Given a sequence $X = (x_1, x_2, x_3)$, we can find $P_{BERT}(x_2 | X_{masked})$ by masking x_2 and calculating the probability with which the model predicts x_2 for the $mask$ token. As mentioned, perplexity calculated in this manner is usually called pseudo-perplexity.

Alternatively, likelihood for masked language models can be computed without the use of masking, by using the outputs of the unmasked sequence directly:

$$L_{BERT}(x_i, X) = P_{BERT}(x_i|X)$$

This alternative method may not seem as intuitive for models pretrained on masked language modelling, as for such models we are often interested in predictions for masked tokens, of which we do not have any in this case. It is, however, relevant when fine-tuning the model for downstream tasks, as mask tokens are usually no longer used, meaning that during fine-tuning, the model works with unmasked sequences and tokens and their likelihoods. Not using masks results in much lower perplexity scores because of the generally very high likelihood of each token due to the model's training objective. This is not an issue, as perplexity scores are compared only within a method, not between them.

For the GPT variants being investigated, likelihoods are computed in a similar fashion, with the difference that likelihood is the probability of the token given (only) its left side context:

$$L_{GPT}(x_i, X) = P_{GPT}(x_i|(x_1, ..., x_{i-1}))$$

To use the same example as above, where we calculate the likelihood of the second token, : $L_{GPT}(x_2, X) = P_{GPT}(x_2|x_1)$
Note that because the sequences we compare to each other differ only in their verbs, which are on the far right of the sequence, the likelihoods of tokens further left do not depend on the differing part of the sequence. Thus, the likelihood of any token left of the verb's tokens in the transitive sequence is equal to its counterpart in the intransitive sequence. This means that by comparing perplexity, we effectively compare likelihoods of the tokens representing the verbs and their right-side context (which in this case is limited to the sentence-ending punctuation mark).

The probabilities put out by the ELECTRA model have a slightly different interpretation from those from other models. Due to not being trained to predict tokens, but rather to distinguish between tokens from the original data and tokens that have been replaced by a generator, ELECTRA outputs the probability $P_{ELECTRA}(x_i|X)$

of each token x_i being an imposter, given the sequence X. The complementary probability $1 - P_{ELECTRA}(x_i|X)$ then represents the probability of a token belonging to the original sequence. I will use this complementary probability as the likelihood of token x_i:

$$L_{ELECTRA}(x_i, X) = 1 - P_{ELECTRA}(x_i|X)$$

Since in training, the probability of a token being an imposter was less than 0.15, average likelihoods are generally greater than 0.85, which leads to very low perplexity compared to other models. As we only compare perplexity within, not between models, this should not be an issue, since perplexity ratio between two alternative sentences, if calculated for the same model, still remains an indication for which out of the two sequences is more likely.

The methods used to calculate perplexity introduced above take perplexity of all tokens of a sequence into account. A different approach is to only consider the likelihoods of tokens referring to the verb of the sentence. Formally, this approach is described by a slightly altered formula:

$$PP_{verb}(X) = \sqrt[l]{\prod_{i=k}^{k+l} \frac{1}{L_{model}(x_i, X)}}$$

where k is the index of the first token of the verb, and l is the number of tokens representing it. As the verb is the only part that is different between the intransitive and transitive sentences, it supposedly holds a big influence on the resulting difference in perplexity as well. Thus, a high perplexity of the verb can be interpreted as the model considering it a bad fit for the context. The idea is that by disregarding likelihoods of tokens in the context, this method avoids some co-occurrence biases models might have learned, resulting in less noisy results. One not very obvious drawback of this method is that it requires every token to be either part of the verb or part of the context. This essentially prevents the Sentence-Piece-based tokenizers of XLM-R, GPT-2, and GPTneo from using this method. For GPT-2 small, basing predictions on verb perplexity is possible, but unnecessary: Since the context to the left of the verb is identical for both sequences that are being compared, and since generative models do not attend to right hand context, the likelihoods of the contexts left of the verbs are the same. The verbs' likelihoods are the first likelihoods

	Sequence	Sequence-Masked	Verb-Only	Verb-Only-Masked
BERT	+	+	+	+
cBERT	+	+	+	+
mBERT	+	+	+	+
ELECTRA	+		+	
XLM-R	+	+		
GPT-2	+			
GPT-2 small	+		+*	
GPTneo	+			

Table 1: model-method-compatibility

that differ between the transitive and intransitive sequence, and as such, are the deciding factor for a prediction based on which sequence has lower perplexity. If a transitive sequence has a higher verb perplexity than its intransitive counterpart, then it will also have higher perplexity across the whole sequence, and vice-versa.

Just like in the case of calulating perplexity for the whole sequence described above, for masked language models, we have the option of using or forgoing masks to calculate likelihoods. The methods compatible with any given model are shown in Table 1.

5 Results

In this section, the results of the experiments across the generated datasets, language models, and perplexity calculation methods are shown.

I consider model accuracy under differing conditions as well as the agreement between models and finally the coefficients that influence model behavior, generally focusing on each model's best performing perplexity calculation mode. The main performance metric is the ratio of transitive predictions on obligatorily transitive sentences. The ratio of transitive predictions on ambiguous sentences, which could take either a transitive or intransitive verb, also needs to be considered, as a model

that always prefers the transitive verb would have an accuracy of 1 on the first metric, without requiring an understanding of when which type of verb is appropriate.

The overall results for model accuracy can be seen in Table 2. Cells are color-coded as follows: Scores below 50% are marked in red, a yellow background signifies a score at about 65% accuracy, and a score of 80% is signified by a green background. One very noticable feature of this table is the poor performance of mBERT on this task. Even with its most favorable mode of perplexity calculation, it barely surpasses random guessing on the base dataset. Curiously, it manages a comparatively strong 70.01% accuracy in verb-masked perplexity mode. This is likely due to some general bias towards transitive verbs in the set of rare verb pairs, as a jump in the ratio of transitive predictions (from 31.78% for the base set to 66.78% given the rare set) can also be observed on ambiguous sentences. Given this poor performance, mBERT will no longer be considered in further analysis.

Table 3 shows the percentage of ambiguous sentences where the model prefers the transitive variant. Color coding goes from green at 10% to yellow at 35% to red at 60%. Here, lower values are desirable, as high values may indicate that a model performs well based on a general pro-transitive bias, instead of grammatical understanding.

5.1 Rare Verbs

The rare verbs dataset was expected to negatively impact the accuracy of predictions due to models having fewer opportunities to learn the transitivity of a verb. This expectation holds true for the two BERT models, whose training dataset is constrained to an older and thus smaller Wikipedia dump, and which therefore had the least exposure to rare words. While the effect of rare verbs on predictions is modest, decreasing accuracy by around two percentage points for each model's respective best perplexity mode, this indicates that more training data could have a positive effect on grammatical understanding, allowing the models to better recognize verb types. For other models using BERT architecture, the effect is less clear. In the case of both ELECTRA and XLM-R, whether rare verbs increase or decrease accuracy

Model	base	rare	longer	rel. cl.
BERT masked	70.79	68.69	74.32	63.51
BERT unmasked	69.34	65.73	65.31	59.05
BERT verb masked	66.49	59.36	65.78	56.04
BERT verb unmasked	61.51	62.66	55.20	53.90
cBERT masked	65.54	66.48	63.44	63.97
cBERT unmasked	71.42	69.74	66.96	67.93
cBERT verb masked	59.19	63.22	56.40	57.32
cBERT verb unmasked	67.12	65.67	60.54	63.23
mBERT masked	52.47	56.72	50.58	49.33
mBERT unmasked	51.07	43.42	57.29	46.80
mBERT verb masked	48.82	70.01	50.45	48.39
mBERT verb unmasked	41.11	60.37	37.85	36.77
ELECTRA	73.83	74.49	74.37	70.00
ELECTRA verb	71.80	70.46	68.50	71.33
XLM-R masked	69.71	73.34	63.50	60.50
XLM-R unmasked	71.58	61.22	78.52	65.21
GPT-2	56.94	65.11	57.10	55.59
GPT-2 small	58.53	61.14	54.93	54.04
GPTneo	61.71	63.49	58.87	57.09

Table 2: Accuracy across datasets in percent.

Model	base	rare	longer	rel. cl.
BERT masked	25.13	43.78	16.89	20.85
BERT unmasked	28.52	47.85	25.11	27.85
BERT verb masked	18.38	49.24	11.42	17.05
BERT verb unmasked	19.43	46.99	09.13	17.20
cBERT masked	24.10	39.59	08.68	19.79
cBERT unmasked	43.73	49.71	32.88	44.29
cBERT verb masked	28.04	47.78	13.70	27.85
cBERT verb unmasked	40.77	48.94	21.92	35.16
ELECTRA	43.87	40.44	32.88	44.60
ELECTRA verb	47.91	42.27	28.31	53.88
XLM-R masked	35.32	51.39	26.03	27.09
XLM-R unmasked	57.59	52.00	59.36	50.53
GPT-2	42.92	44.87	42.92	45.51
GPT-2 small	17.99	38.14	12.33	15.98
GPTneo	40.81	39.29	38.81	41.70

Table 3: Percentage of transitive predictions for ambiguous sentences.

compared to the base set seems to depend on the the perplexity mode. In the case of XLM-R, masked mode in fact gains some accuracy in this situation, while losing over 10% in masked mode. Both modes show the same respective tendency for ambiguous sentences, indicating that at least some of the difference between common and uncommon verb pairs is motivated by a different general bias, rather than a lack of knowledge about uncommon pairs. Given that XLM-R was trained on almost two orders of magnitude more data, this is unsurprising. What is confusing, however, is that these tendencies move in opposite directions: While masked mode tends towards transitive predictions given rarer verbs, unmasked mode strongly prefers intransitive predictions for the same verb pairs. This is not the only instance of XLM-R behaving in an unexpected manner, as will be discussed later. It is possible that the model attends to some features that have not been considered.

For ELECTRA, results for rare verb pairs remain relatively stable. While ELECTRA is trained on a more current and therefore bigger version of the Wikipedia dataset, the size of this training data is still similar to that of the two BERT models. As such, its consistent performance on rare verbs suggests a solid understanding of verb types and therefore supports the claims of drastically improved training efficiency due to its pretraining objective made by Clark et al. (2020).

An interesting phenomenon can be observed in the three GPT-based models. For GPT-2, and to a lesser extend GPT-2 small and GPTneo, less frequent verbs seem to have a positive effect on transitive predictions on obligatorily transitive sentences, where only for GPT-small shows by an increase in transitive predictions on ambiguous sentences. For the other two models, this behavior suggests that their knowledge of the verbs' transitivity is not the limiting factor for performance on this task. This is especially pronounced for GPT-2, which gained around 8% accuracy, while transitive predictions on ambiguous sentences increased by only roughly 2%. While much weaker, the same effect is seen for GPTneo. A possible explanation is that given the fairly large training sets of these models, they have no issues learning transitivities even for rare verbs, while for common verbs frequency and co-occurrence bias start becoming more relevant to the models.

The generally observable trend is that rare verbs do not have a very strong impact

on accuracy, but do (at times sharply) increase transitive predictions on ambiguous sentences, especially for models that are otherwise very good at predicting possible intransitive sentences as intransitive.

5.2 Relative Clauses

The introduction of relative clauses affects phrases in multiple ways: It increases sequence length, and makes sentence structures more complicated, by adding additional arguments not dependent on the main verb of the sentence and possibly confusing particles. Comparing effects on the *relative clauses* dataset to those on the *longer* dataset should in theory allow us to isolate the impact increased complexity has from effects that are merely due to sequence length, as longer sentences are designed to increase the latter without significantly increasing the former.

Results draw an interesting picture: One might have expected to observe a tendency to more frequently assign transitive verbs to longer sentences, as such sentences are, on average, longer, given the additional argument they can accommodate. However, this generally does not hold true for the models under inspection.

Apart from BERT and XLM-R, most models, when using their best perplexity modes, seem to perform somewhat similar for longer and relative clauses. This implies that these models do in fact recognize these structures relatively reliably, as a negative effect on performance is more likely a result of increased length than complexity. For XLM-R, here we find another example of strange behavior: Its unmasked mode accuracy on the longer dataset looks quite impressive, but is somewhat put into perspective when considering it is strongly biased towards transitive predictions, as is demonstrated by its behavior on ambiguous sentences. In masked mode, it shows much less bias, but also loses around 10% accuracy on longer sequences. BERT loses a similar amount of accuracy when given relative clause sentences. While it seems to handle longer sequences quite well, the increased complexity introduced with relative clauses seems to have a particularly strong effect on this model. Interestingly, the same observation can not be made with cBERT, which in unmasked mode loses some accuracy for longer sentences in general, but deals surprisingly well

with the increased complexity of relative clauses. ELECTRA, while showing some loss of accuracy likely as a result of more complex sentence structure in its base perplexity mode, retains the title as the most accurate model.

Lastly, none of the three GPT-based models display a strong behavioral difference between longer- and relative-clause-sentences, and only GPT-2 small seems to be affected by sequence length. However, given their poor baseline accuracy scores despite their (apart from GPT-2 small) relatively strong bias towards transitive verbs, their performance does not measure up to their BERT-based counterparts.

5.3 Causal models

As mentioned, the generative models in this comparison seem to under-perform. There is some evidence to suggest that the Sentence-Piece tokenizer used by GPT-2 and GPTneo is partly to blame for this shortcoming: GPT-2 small, while having less than a third of the parameters that GPT-2 has, still outperforms it on almost every metric. Given that the defining differences between the two models are their tokenizers and size, and given that size appears to have a significant positive effect on performance, as seen in the comparison between GPT-2 and GPTneo, it stands to reason that a model of the size of GPTneo using GPT-2 small's tokenizer may very well reach competitive performance.

5.4 Bidirectional Models

The models based on BERT architecture display a wide variety of behaviors. While BERT performs comparatively well for most tasks, but seems to get confused by relative clauses, cBERT handles those much better than expected, instead losing some performance for longer sequences in general. ELECTRA demonstrates consistently high accuracy, implying a good understanding compared to other models, but also has a stronger bias towards transitive verbs, especially in verb perplexity calculation mode. A similar but even more pronounced behavior can be seen in XLM-R in unmasked mode: Here, ambiguous sentences consistently have a higher than 50%

chance to be predicted as transitive. This should be considered a detriment, since in the Wikipedia dataset, the ratio of transitive to intransitive verbs from the set of common verb pairs is roughly 1.

All in all, while all of the bidirectional models in this test (apart from mBERT) outperform the three unidirectional models (at least in their best modes), they display a surprising amount of variance in their strengths and weaknesses.

5.5 Perplexity Modes

The method of calculating perplexity (Using masks versus not using masks, considering only the verb's likelihood versus considering that of the whole sequence) is an influential factor for the quality of predictions, which needs to be evaluated on a case-by-case basis for each model.

Given that Transitivity Prediction revolves around deciding the more appropriate verb, a reasonable assumption to make is that most of the acceptability difference in each sequence can be found in this verb. The likelihoods of other tokens may introduce noise by interacting with other tokens, unrelated to transitivity. Avoiding this noise is the motivating idea for the verb perplexity mode. However, the only model with which this mode produces accuracies that are comparable to those of other modes of the same model is ELECTRA. This implies that in the case of ELECTRA, most of thee difference in perplexity between the sentences does in fact stem from the verb. For obligatorily transitive sentences, this seems to hold true most of the time, as in such situations, an intransitive verb is generally correctly considered as less likely. For ambiguous sentences, however, tokens other than the verb seem to lose some likelihood if the verb is transitive, resulting in higher sequence perplexity compared to verb perplexity. A different effect can be observed with BERT: regardless of masking, the verb itself displays a bias towards the intransitive prediction, when compared to the rest of the sentence. For the most part, the same holds true for cBERT.

The approach of calculating likelihoods of tokens without masks was proposed by Salazar et al. (2019) as a trade-off between computational demand and accuracy.

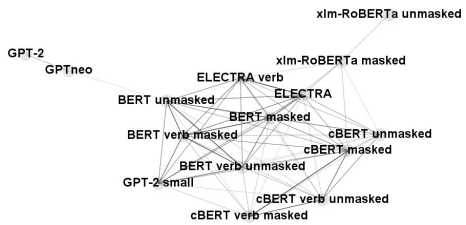

Figure 1: Agreement graph of base dataset

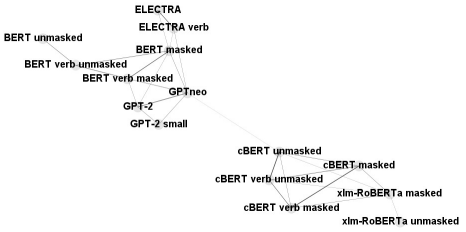

Figure 2: Agreement graph of rare verbs dataset

It also resembles the situation for most downstream tasks that do not implement a mask token. The unmasked mode was therefore not expected to perform as well as the masked mode. This expectation is confirmed by BERT, but not my cBERT, resulting in BERTs best-performing perplexity mode being the masked mode, while for cBERT, it is the unmasked mode. I argue that in spite of lower accuracy in some situations, ELECTRA's base mode and XLM-R's masked mode should be considered the best modes of their respective models, since their alternative modes display extreme transitive bias which can be considered problematic as described above.

5.6 Agreements

To understand how these models differ in their language understanding, it may be helpful to see what models produce similar predictions. Attending to similar features of a sentence should logically produce similar results. Tables showing agreements between all the models for each dataset can be found in appendix A.

A general trend that emerges is that, unsurprisingly, different modes of the same model tend to agree strongly with each other. For example, BERT's average within-model agreement across all modes for the base dataset is above 80%, which is higher than its agreement with any other model. High agreement within models can be observed for all tested models. Another observation is that both of the big GPT-based models agree the most with each other. While their absolute agreement value is not as high as that between other models, their agreement with others is even

lower. Interestingly, GPT-2 small does not seem to behave very similar to these two models, instead agreeing most strongly with BERT's "verb" modes. On one hand, this implies that GPT-2 and GPTneo behave differently due to their size, their tokenizer, or a combination of both. Agreement with BERT verb on the other hand is not very surprising when one considers how similar the pretraining tasks of each model become in this situation: Predicting a masked verb token at the end of a sentence and predicting the next token in a sentence that only requires a verb are intuitively closely related tasks.

Figures 1 and 2 are an attempt to visualize the different families of models that show similar behavior. To create these visualizations, first, a fully connected graph with models as nodes and their agreements as edge weights was created. Then, starting with the lowest edge weight, edges where removed until the next edge removal would disconnect the graph. Node positions were then adjusted according to the ForceAtlas2 algorithm (Jacomy et al., 2014).

Figure 1, which represents agreements on the base dataset, shows that models effectively create three clusters: The main cluster contains BERT-based monolingual models, as well as GPT-2 small, with modes of the same models usually being closer together. Apart from this cluster, there is a cluster made up of the two bigger GPT-type models, and lastly, the cluster of XLM-R. GPT-2 and GPTneo forming a distinct group is expected, as these models already demonstrated some fundamental differences. XLM-R's distinct behavior has two possible explanations: Its SentencePiece-based tokenizer is the chief suspect, but its multilingual training may also be causing it to process language in a different way from its monolingual counterparts. The clustering for the longer and relative clauses dataset draws a similar picture for the GPT cluster, while the XLM-R cluster moves towards the center cluster.

However, different behavior can be observed for the rare verbs dataset (Figure 2). Here, of the two clusters that form, one consists of cBERT and XLM-R, while the other includes both the GPT- and the remaining BERT-based models. GPT models behaving differently for rare verbs had already been expected, given the results we saw on their performance. The relatively high agreement with BERT and

ELECTRA suggests that models with GPT architecture consider similar features for verbs where not enough data is present to base decisions on co-occurrence. With increasing familiarity with the verb and its usual context, agreement starts to drift apart. The high level of agreement between cBERT and XLM-R for this dataset is hard to explain: As these models differ in tokenization, training set, and to a lesser extent, even pretraining task, cBERT agreeing with XLM-R, as opposed to, for example, the on paper much more similar BERT, seems almost arbitrary.

5.7 Logistic regression

To find out what sentence features inform the decision of a model, I fit a regression model to its predictions. For this, i use the R library *glmnet* (Friedman et al., 2010), which uses Elastic-Net-regression by default. I utilize ten-fold cross validation. The considered independent variables are the identity of the verb pair, the constituent order (meaning the order in which arguments and adverbials are prefixed to the verb), the identity and marker of the first and second argument, the adverbial used, and in the case of the relative clauses dataset, the position and type of relative clause added. The value of these coefficients tell us how strongly each of these features correlates with a transitive or intransitive prediction.

Across all models and modes, there are 181 coefficients to consider, plus 7 more for relative clauses. Ideally, those coefficients that indicate an argument with a *wo* particle should have a high positive values, indicating high correlation with transitive predictions. The same goes for coefficients indicating the presence of multiple arguments. Factors such as verb pair identity, adverbial identity, and type and position of relative clauses should not play too big of a role. Tables 4, 5, 6, and 7 show the maximum and minimum coefficients for BERT masked and GPT-2. [11]

[11]The four coefficient tables resulting from this regression are available at these links: `https://drive.google.com/file/d/15IaFkOiyo0Q85Vk2CKSCgs8UwP36YrBo/view?usp=sharing`, `https://drive.google.com/file/d/1XeL1VdrN7ADr4PGjQSehYaEV-6jMvPFT/view?usp=sharing`, `https://drive.google.com/file/d/1pn4M8gZj5qsZuo1IiTgJJd6nQBhZOo9X/view?usp=sharing`, `https://drive.google.com/file/d/1t_m8AKtvtnb5_Qs-G7mhVAv3J1Xh2G3C/view?usp=sharing`

marker1w	2.591735947
arg1 id1.marker1g	2.278545687
marker2w	2.246562109
arg1 id2.marker1g	1.767797672
verb pair id286	1.507920641
COwgav	1.371698718
verb pair id265	1.361102779
arg2 id1.marker2g	1.270775074
arg1 id4.marker1g	1.265723244
arg2 id2.marker2g	1.216086325

Table 4: Max. coefficients for BERT on *base*

verb pair id141	-4.459019510
arg1 id10.marker1g	-2.562459711
arg1 id8.marker1g	-2.534395423
verb pair id153	-2.512351104
COagv	-2.425492253
arg1 id7.marker1g	-1.994570176
COgv	-1.984526274
verb pair id282	-1.712175691
verb pair id296	-1.524854822
arg1 id11.marker1w	-1.514078384

Table 5: Min. coefficients for BERT on *base*

X.Intercept.	2.715971e+00
COwav	2.712243e+00
COgwav	2.632784e+00
COhwav	2.622277e+00
arg1 id3.marker1w	1.773101e+00
marker2w	1.693284e+00
verb pair id28	1.665896e+00
arg1 id2.marker1w	1.664022e+00
verb pair id25	1.410473e+00
verb pair id270	1.289851e+00

Table 6: Max. coefficients for GPT-2 on *base*

verb pair id271	-1.024969e+01
verb pair id153	-9.289100e+00
verb pair id129	-6.962328e+00
verb pair id255	-6.494575e+00
verb pair id141	-5.261678e+00
verb pair id281	-4.998784e+00
verb pair id77	-4.334255e+00
verb pair id140	-4.236711e+00
verb pair id217	-4.175004e+00
verb pair id150	-4.169882e+00

Table 7: Min. coefficients for GPT-2 on *base*

Particles indicating a direct object and therefore a sentence requiring a transitive verb have strong correlation to transitive predictions across most models and modes. We can observe that such a particle has a stronger impact on the prediction if it is further back in the sentence and therefore closer to the verb. Another feature that shows a strong positive correlation with transitive predictions is a definite personal pronoun followed by the subject marker *ga*. Almost all tested models seem to assume a subject high in agency such as a definite person to be transferring an action to something or someone else, making a transitive verb more likely.

The identity of verb pairs often has a strong negative correlation with transitive predictions for both GPT-2 and GPTneo. In fact, for both of these models, the 20 coefficients most strongly related to intransitive predictions on the base dataset are all verb pair identities. This effect seems to be less severe on the rare verbs dataset, lending credibility to the hypothesis that these models decide based on frequency biases for common verbs. Strong verb bias, as observed here, may also partially explain these models' underwhelming performance. While the two big GPT-based models most strongly exhibit this property, they are by no means the only models with strong intransitive bias for certain verb pairs. All models seem to behave in this manner to some extent. Interestingly, verb perplexity modes seem to consistently rely more on verb identity for their predictions than their whole-sequence-considering counterparts. Furthermore, well-performing modes such as BERT masked, cBERT unmasked, and ELECTRA base, seem to have few verb pairs with high positive coefficients, implying their transitive predictions are usually not due to verb bias.

Relative clauses seem to be recognized surprisingly well by most models. A relative clause containing the object-marking particle *wo* is not correlated with more transitive predictions. In fact, the opposite seems to be true. Relative clauses of any kind have on average an intransitive coefficient. However, the effect of relative clauses is also highly dependent on the position of the relative clause: Relative clauses modifying the argument closer to the verb usually cause a moderate intransitive tendency, while those modifying an argument further left cause a slight transitive tendency on average. This behavior is consistent across models.

Adverbial phrases have no strong correlation with predictions. However, GPT-2

and GPTneo seem to consider certain constituent orders, which often include adverbials, as indicators of transitive sentences. Their emphasis on specific constituent orders indicates that these models are looking for sentences following standard sentence order, which for Japanese is SOV. The constructed datasets do not exclusively use this order, giving another possible explanation for the poor performance of these models.

6 Discussion

In this chapter, I will discuss the findings and limitations of this work.

The goal of this research was to determine the extend to which current state-of-the-art pretrained language models are able learn and apply grammatical rules, as exemplified by the task of Transitivity Prediction. Results show large differences in performance between the best and worst performing models.

The worst performing models, apart from mBERT, were the generative GPT models. There are multiple possible explanations for their results: First and foremost, the SentencePiece-based tokenizer used by both GPTneo and GPT-2 creates faulty tokenizations containing tokens that contain parts of multiple words, making it difficult for the model to assign a specific grammatical role to a token in the way it normally would. This might also make learning grammatical structures migh more difficult. Evidence for this can be seen in the the performance of GTP-2 small compared to GPT-2. Despite its smaller size, the former outperforms the latter, likely because it does not have to contend with a faulty tokenizer. However, when compared to the bidirectional models being analyzed, which all effectively have the same size as GPT-2 small, its performance is still the weakest by a large margin. The explanation for GPT's under-performance is therefore likely to be found in its architecture. Although the task was specifically chosen to create as much of an even playing field as possible between bidirectional and unidirectional models, some differences are hard to compensate for. While the deciding factor, the verb, is at the end of the sentence, giving both architectures access to the complete context,

bidirectional models also have the information that the sequence does in fact end after the verb. This information is not available to unidirectional model, making it theoretically possible for them to try to salvage the grammaticality of a sequence by adding more words.

Furthermore, analysis of regression model coefficients indicates that unidirectional models have much stronger biases due to verb pair identity, a property they share with the verb perplexity modes of unidirectional models. This is somewhat plausible, considering that in both cases, perplexity is based exclusively the verb. In fact, the predictions of some of the verb modes do bare a resemblance to those of GPT-2small in both overall accuracy and labels for individual sentences.

For the bidirectional models, multiple methods of calculating perplexity were explored. Masked models allow for perplexity calculation with and without masking. While not masking tokens is the approach that more closely resembles the reality of most fine-tuning tasks, it is rarely employed when it comes to perplexity calculation. This might be an oversight, as cBERT's results suggest that some models achieve better performance with this perplexity mode. Further research may be required to gain a better understanding of the different strengths and weaknesses of masked and unmasked modes of perplexity calculation.

The second variation in perplexity calculation is to only consider the token(s) under inspection, which may be appropriate depending on the task, and, as mentioned, in some ways resembles the perplexity of unidirectional models. However, this mode ultimately did not reach the performance level of its alternative. A possible reason for this are biases for or against certain verbs. Other tokens such as particles may also have a moderating effect in such cases.

While bidirectional models outperformed unidirectional ones, none of the tested models achieved exceptional results. There are several possible explanations:

- The models do not have the required grammatical understanding.
- The methods used are unable to extract the model's grammatical understanding.

- Flaws in the test set-up introduce noise.

While some results, such as BERT seemingly getting confused by relative clauses, point towards a lack of grammatical understanding, others, like GPT-2's preference for certain sentence structures, suggest that understanding may be present within the model, but may not always be expressed in its behavior. Lastly, within the dataset that is used, most sentences are semantically nonsensical. This may be more of a problem for a language that is highly sensitive to context, like Japanese, than it would be for other languages such as English.

Lastly, the multilingual XLM-R model performed surprisingly well, especially considering its SentencePiece tokenizer and the extremely poor performance of mBERT. An interesting aspect of this model is that it seemingly attends to different parts of sentence structure, compared to monolingual models, as shown by its general disagreement with other models using the same architecture.

7 Conclusion

In this work, I assess grammatical understanding of multiple state-of-the-art Japanese language models based on their ability to distinguish sentences that require a transitive verb from those that do not, using perplexity as a referee. Models selection has the goal of covering various architectures, tokenization strategies, and pretraining regimens. To evaluate the models, I construct four datasets of minimal pairs. The *base* dataset uses frequently occurring verbs, the *rare verbs* dataset uses infrequent verbs, the *longer* dataset adds a neutral phrase at the beginning of sentences, increasing their length without a significant increase in complexity, and the *relative clauses* dataset adds relative clauses to nouns, increasing complexity and length of the sentences.

Where appropriate, I experiment with different methods for calculating perplexity: For masked language models, I calculate perplexity once by sequentially masking tokens and once without masking. For models not using a SentencePiece-based to-

kenizer, I consider perplexity of the verb itself in addition to that of the whole sequence.

I find that not masking tokens, which is, to my knowledge, a strategy rarely employed for perplexity calculation, improves performance of a character-based BERT model on the Transitivity Prediction task. Bidirectional models consistently outperform unidirectional ones, even if they are much smaller. This seems to be in part due to a erroneous tokenization, and in part due to an architectural disadvantage. BERT, cBERT, ELECTRA, and GPT-2 small generally learn to attend to similar features of sentences. XLM-R shows similar performance to other BERT-based models, but seems to have a slightly different understanding of language. Lastly, while there is a large difference between the worst and best performing models, none of them demonstrate exceptional understanding, given this specific task.

A Appendix

A.1 Agreement figures

BERT masked	100	78.63	84.37	77.64	74.19	70.05	66.68	67.45	78.1	76.33	68.94	63.12	60.83	74.6	63.81
BERT unmasked	78.63	100	80.84	84.41	70.84	69.31	65.19	66.37	75.15	73.4	64.38	62.69	60.5	73.47	64.68
BERT verb masked	84.37	80.84	100	83.51	73.17	68.17	68.43	68.05	75.98	75.91	64.28	60.6	61.95	78.62	63.4
BERT verb unmasked	77.64	84.41	83.51	100	72.15	66.23	68.55	68.81	72.44	73.26	60.91	56.66	58.9	78	62.85
cBERT masked	74.19	70.84	73.17	72.15	100	71.13	85.56	78.78	71.75	70.17	67.93	59.09	56.95	70.47	61.1
cBERT unmasked	70.05	69.31	68.17	66.23	71.13	100	66.02	73.4	71.24	68.71	65.28	61.99	58.46	64.98	60.78
cBERT verb masked	66.68	65.19	68.43	68.55	85.56	66.02	100	80.49	64.96	64.21	62.98	54.15	54.9	66.87	56.7
cBERT verb unmasked	67.45	66.37	68.05	68.81	78.78	73.4	80.49	100	66.48	66.24	63.08	56.82	55.16	66.05	56.17
ELECTRA	78.1	75.15	75.98	72.44	71.75	71.24	64.96	66.48	100	86.51	67.79	63.88	59.52	71.76	64.16
ELECTRA verb	76.33	73.4	75.91	73.26	70.17	68.71	64.21	66.24	86.51	100	65.91	62.27	59.7	72.78	63.62
XLM-R masked	68.94	64.38	64.28	60.91	67.93	65.28	62.98	63.08	67.79	65.91	100	67.28	57.85	62.6	63.96
XLM-R unmasked	63.12	62.69	60.6	56.66	59.09	61.99	54.15	56.82	63.88	62.27	67.28	100	58.18	55.68	60.15
GPT-2	60.83	60.5	61.95	58.9	56.95	58.46	54.9	55.16	59.52	59.7	57.85	58.18	100	61.1	69.45
GPT-2 small	74.6	73.47	78.62	78	70.47	64.98	66.87	66.05	71.76	72.78	62.6	55.68	61.1	100	64.04
GPTneo	63.81	64.68	63.4	62.85	61.1	60.78	56.7	56.17	64.16	63.62	63.96	60.15	69.45	64.04	100

Figure 3: Agreements for base dataset

BERT masked	100	75.39	84.01	73.57	74.08	68.51	66.96	67.53	79.32	77.84	68.76	67.14	61.01	71.98	61.37
BERT unmasked	75.39	100	77.04	78.32	70.71	66.34	65.59	66.1	74.07	72.39	62.95	61.15	59.04	71.57	62.48
BERT verb masked	84.01	77.04	100	82.64	74.37	66.22	70.33	69.44	76.95	78.05	65.15	60.83	61.88	79.41	62.46
BERT verb unmasked	73.57	78.32	82.64	100	73.31	62.42	71.71	70.38	69.89	73.51	61.47	52.63	57.72	80.13	60.86
cBERT masked	74.08	70.71	74.37	73.31	100	70.15	84.5	79.49	72.4	72.1	67.5	60.58	57.79	71.5	60.21
cBERT unmasked	68.51	66.34	66.22	62.42	70.15	100	66.21	70.98	69.15	65.55	63.91	64.65	58.34	63.03	59.74
cBERT verb masked	66.96	65.59	70.33	71.71	84.5	66.21	100	81.57	66.44	68.4	62.86	56.36	54.79	70.08	57.09
cBERT verb unmasked	67.53	66.1	69.44	70.38	79.49	70.98	81.57	100	66.15	67.49	63.85	58.85	55.18	68.09	56.2
ELECTRA	79.32	74.07	76.95	69.89	72.4	69.15	66.44	66.15	100	84.52	66.02	67.04	59.75	70.54	61.45
ELECTRA verb	77.84	72.39	78.05	73.51	72.1	65.55	68.4	67.49	84.52	100	65.16	63.56	60.69	74.36	61.15
XLM-R masked	68.76	62.95	65.15	61.47	67.5	63.91	62.86	63.85	66.02	65.16	100	64.12	55.03	62.04	61.25
XLM-R unmasked	67.14	61.15	60.83	52.63	60.58	64.65	56.36	58.85	67.04	63.56	64.12	100	57.39	53.08	57.89
GPT-2	61.01	59.04	61.88	57.72	57.79	58.34	54.79	55.18	59.75	60.69	55.03	57.39	100	61.07	68.26
GPT-2 small	71.98	71.57	79.41	80.13	71.5	63.03	70.08	68.09	70.54	74.36	62.04	53.08	61.07	100	63.77
GPTneo	61.37	62.48	62.46	60.86	60.21	59.74	57.09	56.2	61.45	61.15	61.25	57.89	68.26	63.77	100

Figure 4: Agreements for longer dataset

BERT masked	100	68.26	81.79	75.82	67.32	66.43	61.14	63.22	70.48	69.53	63.67	58.16	70.54	65.94	72.4
BERT unmasked	68.26	100	62.95	74.79	61.71	62.5	59.48	60.07	65.25	62.51	62.79	59.96	59.67	57.17	59.97
BERT verb masked	81.79	62.95	100	76.33	58.96	61.19	54.26	57.96	59.41	62.53	58.77	55.18	71.34	66.37	72.33
BERT verb unmasked	75.82	74.79	76.33	100	62.45	65.31	58.83	62.26	63.69	65.23	62.33	57.89	67.09	64.05	69.12
cBERT masked	67.32	61.71	58.96	62.45	100	74.16	86.03	76.86	67.95	67.24	73.04	62.85	66.78	63.74	67.72
cBERT unmasked	66.43	62.5	61.19	65.31	74.16	100	72.38	87.43	66.97	66.8	69.82	61.97	67.52	64.86	68.39
cBERT verb masked	61.14	59.48	54.26	58.83	86.03	72.38	100	78.52	61.16	62.04	72.16	63.09	64.95	58.8	63.57
cBERT verb unmasked	63.22	60.07	57.96	62.26	76.86	87.43	78.52	100	63.94	64.78	69.87	62.63	67.42	64.31	66.23
ELECTRA	70.48	65.25	59.41	63.69	67.95	66.97	61.16	63.94	100	80.38	66.86	59.85	64.64	64.58	66.68
ELECTRA verb	69.53	62.51	62.53	65.23	67.24	66.8	62.04	64.78	80.38	100	66.69	60.47	66.85	65.54	71.36
XLM-R masked	63.67	62.79	58.77	62.33	73.04	69.82	72.16	69.87	66.86	66.69	100	69.69	64.89	62.42	65.89
XLM-R unmasked	58.16	59.96	55.18	57.89	62.85	61.97	63.09	62.63	59.85	60.47	69.69	100	58.13	55.48	58.02
GPT-2	70.54	59.67	71.34	67.09	66.78	67.52	64.95	67.42	64.64	66.85	64.89	58.13	100	72.5	80.28
GPT-2 small	65.94	57.17	66.37	64.05	63.74	64.86	58.8	64.31	64.58	65.54	62.42	55.48	72.5	100	72.22
GPTneo	72.4	59.97	72.33	69.12	67.72	68.39	63.57	66.23	66.68	71.36	65.89	58.02	80.28	72.22	100

Figure 5: Agreements for rare verbs dataset

A.2 Zusammenfassung auf Deutsch

Moderne auf Aufmerksamkeitsmechanismen basierende Sprachmodelle wie BERT und GPT zeigen bessere Ergebnisse in vielen NLP-Aufgaben, als die Modelle, die bis dahin den Stand der Technik verkörpert hatten. Derartige Ergebnisse implizieren einen Grad von Verständnis von grammatikalischen Strukturen. Diese Arbeit erkundet die Fähigkeit von Sprachmodellen, Transitivität von Verben auf Japanisch vorherzusagen, und versucht so, einen Beitrag zu der wachsenden Menge an Forschung an solchem Sprachverständnis zu leisten. Ich vergleiche eine Vielzahl verschiedener Sprachmodelle mit unterschiedlichen Architekturen, Tokenisierungsansätzen, Trainingsregimenten und -datensätzen. Hierdurch finde ich, dass bidirektionale Modelle generell bessere Ergebnisse erzielen als Unidirektionale, und dass verschiedene Me-

BERT masked	100	76.22	82.71	78.23	76.58	68.6	68.43	68.21	77.2	75.87	67.44	61.75	60.85	76.28	61.99
BERT unmasked	76.22	100	78.46	82.88	71.01	66.06	65.71	65.7	71.13	70.65	61.86	59.87	58.65	73.81	60.77
BERT verb masked	82.71	78.46	100	86.65	74.24	64.59	69.97	68.29	73.57	74.27	63.71	58.38	61.52	81.29	62.16
BERT verb unmasked	78.23	82.88	86.65	100	72.84	63.61	69.12	68.89	70.94	72.6	61.61	55.68	58.2	80.63	60.87
cBERT masked	76.58	71.01	74.24	72.84	100	73.32	83.97	78.78	73.64	72.1	69.65	60.25	57.36	72.71	60.66
cBERT unmasked	68.6	66.06	64.59	63.61	73.32	100	67.78	74.86	70.59	68.94	64.01	60.6	58.52	64.61	59.48
cBERT verb masked	68.43	65.71	69.97	69.12	83.97	67.78	100	81.4	65.82	65.22	66.5	55.29	54.23	69.05	56.43
cBERT verb unmasked	68.21	65.7	68.29	68.89	78.78	74.86	81.4	100	66.75	67.42	64.94	57.27	56.29	68.25	55.61
ELECTRA	77.2	71.13	73.57	70.94	73.64	70.59	65.82	66.75	100	85.18	65.76	62.04	59.88	71.82	61.97
ELECTRA verb	75.87	70.65	74.27	72.6	72.1	68.94	65.22	67.42	85.18	100	64.81	62.2	60.79	72.72	61.55
XLM-R masked	67.44	61.86	63.71	61.61	69.65	64.01	66.5	64.94	65.76	64.81	100	64.99	56.24	63.01	61.77
XLM-R unmasked	61.75	59.87	58.38	55.68	60.25	60.6	55.29	57.27	62.04	62.2	64.99	100	56.4	56.04	56.99
GPT-2	60.85	58.65	61.52	58.2	57.36	58.52	54.23	56.29	59.88	60.79	56.24	56.4	100	61.18	69.6
GPT-2 small	76.28	73.81	81.29	80.63	72.71	64.61	69.05	68.25	71.82	72.72	63.01	56.04	61.18	100	62.72
GPTneo	61.99	60.77	62.16	60.87	60.66	59.48	56.43	55.61	61.97	61.55	61.77	56.99	69.6	62.72	100

Figure 6: Agreements for relative clauses dataset

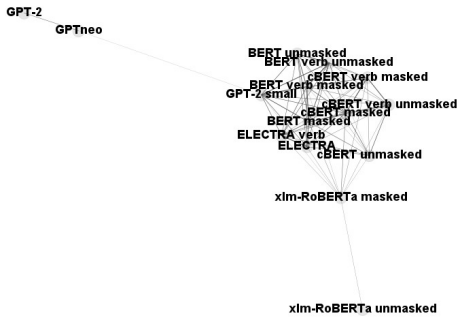

Figure 7: agreement graph of rel. clauses dataset

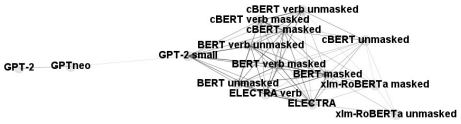

Figure 8: agreement graph of longer dataset

thoden der Perplexitätsberechnung situationsbedingt vorteilhaft sein können. Außerdem schließe ich, dass die getesteten Modelle ein nur lückenhaftes Verständnis für die Grammatik erlangt haben, die für Transitivitätsvorhersage notwendig ist.

References

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. Exploring length generalization in large language models. *arXiv preprint arXiv:2207.04901*, 2022.

Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu, and Yunhai Tong. Syntax-bert: Improving pre-trained transformers with syntax trees. *arXiv preprint arXiv:2103.04350*, 2021.

Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. On the ability and limitations of transformers to recognize formal languages. *arXiv preprint arXiv:2009.11264*, 2020.

Jim Breen. JMdict: a Japanese-multilingual dictionary. In *Proceedings of the Workshop on Multilingual Linguistic Resources*, pages 65–72, Geneva, Switzerland, August 28 2004. COLING. URL `https://aclanthology.org/W04-2209`.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.

Bernard Comrie and Maria Polinsky. *Causatives and transitivity*, volume 23. John Benjamins Publishing, 1993.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116, 2019. URL `http://arxiv.org/abs/1911.02116`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018. URL `https://arxiv.org/abs/1810.04805`.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010. doi: 10.18637/jss.v033.i01. URL `https://www.jstatsoft.org/v33/i01/`.

Yoav Goldberg. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*, 2019.

Martin Haspelmath. More on the typology of inchoative/causative verb alternations. *Causatives and transitivity*, 23:87–121, 1993.

John Hewitt and Christopher D Manning. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, 2019.

Paul J Hopper and Sandra A Thompson. Transitivity in grammar and discourse. *language*, pages 251–299, 1980.

Jennifer Hu, Jon Gauthier, Peng Qian, Ethan Wilcox, and Roger P Levy. A systematic assessment of syntactic generalization in neural language models. *arXiv preprint arXiv:2005.03692*, 2020.

Mathieu Jacomy, Tommaso Venturini, Sebastien Heymann, and Mathieu Bastian. Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. *PloS one*, 9(6):e98679, 2014.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1356. URL `https://aclanthology.org/P19-1356`.

Taro Kageyama and Wesley M Jacobsen. *Transitivity and valency alternations: Studies on Japanese and beyond*, volume 297. Walter de Gruyter GmbH & Co KG, 2016.

Taeuk Kim, Jihun Choi, Daniel Edmiston, and Sang-goo Lee. Are pre-trained language models aware of phrases? simple but strong baselines for grammar induction. *arXiv preprint arXiv:2002.00737*, 2020.

Konstantinos Kogkalidis and Gijs Wijnholds. Discontinuous constituency and bert: A case study of dutch, 2022. URL `https://arxiv.org/abs/2203.01063`.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. Discourse probing of pretrained language models. *arXiv preprint arXiv:2104.05882*, 2021.

Taku Kudo. Mecab: Yet another part-of-speech and morphological analyzer. *http://mecab. sourceforge. net/*, 2005.

Nayeon Lee, Yejin Bang, Andrea Madotto, Madian Khabsa, and Pascale Fung. Towards few-shot fact-checking via perplexity. *arXiv preprint arXiv:2103.09535*, 2021.

Yongjie Lin, Yi Chern Tan, and Robert Frank. Open sesame: getting inside bert's linguistic knowledge. *arXiv preprint arXiv:1906.01698*, 2019.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. Linguistic knowledge and transferability of contextual representations. *arXiv preprint arXiv:1903.08855*, 2019a.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019b.

Samuel E Martin. *A reference grammar of Japanese*. University of Hawaii Press, 2003.

Rebecca Marvin and Tal Linzen. Targeted syntactic evaluation of language models. *arXiv preprint arXiv:1808.09031*, 2018.

Alessio Miaschi, Dominique Brunato, Felice Dell'Orletta, and Giulia Venturi. What makes my model perplexed? a linguistic investigation on neural language models

perplexity. In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 40–47, 2021.

Hiroshi Minashima. On the deletion of accusative case markers in japanese. *Studia Linguistica*, 55(2):176–191, 2001.

Aaron Mueller, Yu Xia, and Tal Linzen. Causal analysis of syntactic agreement neurons in multilingual language models, 2022. URL `https://arxiv.org/abs/2210.14328`.

Pedro Javier Ortiz Suarez, Laurent Romary, and Benoit Sagot. A monolingual approach to contextualized word embeddings for mid-resource languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1703–1714, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.156. URL `https://aclanthology.org/2020.acl-main.156`.

Matthew E Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. Dissecting contextual word embeddings: Architecture and representation. *arXiv preprint arXiv:1808.08949*, 2018.

Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*, 2019.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training (2018), 2018.

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

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020.

Julian Salazar, Davis Liang, Toan Q Nguyen, and Katrin Kirchhoff. Masked language model scoring. *arXiv preprint arXiv:1910.14659*, 2019.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

Yixiao Song, Kalpesh Krishna, Rajesh Bhatt, and Mohit Iyyer. Sling: Sino linguistic evaluation of large language models, 2022. URL `https://arxiv.org/abs/2210.11689`.

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures. In *7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7)*. Leibniz-Institut für Deutsche Sprache, 2019.

Wilson L Taylor. "cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433, 1953.

Nobuhiro Ueda, Daisuke Kawahara, and Sadao Kurohashi. BERT-based cohesion analysis of Japanese texts. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1323–1333, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.114. URL `https://aclanthology.org/2020.coling-main.114`.

Masato Umakoshi, Yugo Murawaki, and Sadao Kurohashi. Japanese zero anaphora resolution can benefit from parallel texts through neural transfer learning. In

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 1920–1934, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.165. URL https://aclanthology.org/2021.findings-emnlp.165.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. URL https://arxiv.org/abs/1706.03762.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Yequan Wang, Jiawen Deng, Aixin Sun, and Xuying Meng. Perplexity from plm is unreliable for evaluating text quality. *arXiv preprint arXiv:2210.05892*, 2022.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R Bowman. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392, 2020.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. Ccnet: Extracting high quality monolingual datasets from web crawl data. *arXiv preprint arXiv:1911.00359*, 2019.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. CCNet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France,

May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://www.aclweb.org/anthology/2020.lrec-1.494.

Shijie Wu and Mark Dredze. Are all languages created equal in multilingual bert? *arXiv preprint arXiv:2005.09093*, 2020.

Beilei Xiang, Changbing Yang, Yu Li, Alex Warstadt, and Katharina Kann. CLiMP: A benchmark for Chinese language model evaluation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2784–2790, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.242. URL https://aclanthology.org/2021.eacl-main.242.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

Erklärung (Statement of Authorship)

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein. [12]

(Gerhard Christian Breul)

[12] translation for convenience: I hereby declare that the work presented in this thesis is entirely other sources and references that the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted hard copies.